Bergische Universität Wuppertal

Multiobjective Optimization of Shapes Using Scalarization Techniques

Dissertation
zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften (Dr. rer. nat.)
an der
Fakultät für Mathematik und Naturwissenschaften
Fachgruppe Mathematik und Informatik

vorgelegt von
Johanna Schultes

betreut durch
Prof. Dr. Kathrin Klamroth

Wuppertal, Juli 2024

Gutachter: Prof. Dr. Kathrin Klamroth
Prof. Dr. Hanno Gottschalk

Acknowledgements

The work has received funding from the BMBF collaborative research project GivEn under the grant no. 05M18PXA and it was also supported by the Centre for Graduate Studies (ZGS) of the University of Wuppertal with a final scholarship.

First of all, I want to thank my supervisor Prof. Dr. Kathrin Klamroth. Thank you so much for your support and guidance on my academic journey. Your help has been invaluable to me. I am truly grateful for everything you have done to guide me along the way, both as a mentor and as a role model.

I also want to thank the whole team from the given project. I very much appreciate Prof. Dr. Matthias Bolten and Dr. Camilla Hahn who played a decisive role in providing my with the necessary code framework. Especially helpful to me during this time were Dr. Jan Backhaus and Dr. Daniel Luft, with whom I spent a lot of time in code sessions. Thanks also to thank Prof. Dr. Hanno Gottschalk and Marco Reese. Without you and the support of the whole team this work would not has been possible. I really enjoyed working with you.

Furthermore, I want to thank my working group for the pleasant atmosphere, the scientific exchange, the coffee breaks and team events. Many thanks to Britta, Michael, Claudia, Onur, Julius, Fabian, David, Lara, Julia, Tobias, Marco, Konstantin and Daniela.

I also wish to thank my friends who have accompanied me through my studies. Special thanks to Linda, Robin, Pascal, Daniel, Lisa, Nathanael and Henrik.

Last but not least, I'm extremely grateful to my family who have always supported and encouraged me. Special thanks to Dennis how never let me down and always brought me coffee and food.

Of course, I would also like to thank all the others who have accompanied me along the way but have not been specifically mentioned here.

Contents

Chapter 1

Introduction

Optimization problems arise in almost all areas of life and span disciplines such as science, economics, industry and technology. Among these, shape optimization problems are widespread in numerous areas, including but not limited to engineering, manufacturing or architecture.

In challenges involving shape optimization, the ideal design of a shape is sought in order to optimize the objective function under consideration.[1] Improving the exterior design of vehicles, aircraft wings, or rocket nozzles to boost performance, cut weight, or lower cost are just a few example for problems that can be considered. In order to evaluate the specific objective the shape is often subject to constraints posed by partial differential equations (PDEs). For instance, the heat equation can be used to describe heat diffusion inside the shape, or the Navier-Stokes equations describe a fluid motion surrounding the shape. This usually requires the use of finite element analysis. Those PDEs as well as geometrical constraints frequently place restrictions on the optimization problems in shape optimization. The versatility of shape optimization makes it a valuable tool for improving performance and functionality across a wide range of applications.

Since the decision-making process often involves multiple competing objectives, multiobjective optimization becomes essential.[2] For instance, while designing and building a bridge, considerations such as the total cost and mechanical integrity must be addressed. A well-known concept for multiobjective optimization problems is *Pareto optimality* named after Vilfredo Pareto[3], where we call a solution of a

[1] See Haslinger and Mäkinen, 2003; Sokolowski and Zolesio, 1992, for example.
[2] A general introduction can be found in Ehrgott, 2005; Miettinen and Mäkelä, 1996.
[3] Vilfredo Pareto, ⋆1848, †1923.

multiobjective optimization problem Pareto optimal or *efficient* if no objective can be improved without compromising at least one of the other objectives. Thus, the aim of multiobjective optimization is often to provide the decision maker with a set of Pareto optimal solutions, enabling informed choices.

One approach to solve multiobjective optimization problems is the use of scalarization methods.[4] The idea of scalarization-based methods is to combine the individual objective functions into one objective function plus possibly adding additional constraints. In this thesis we focus on scalarization methods that do not add further constraints to the optimization problem. With the weighted sum method, for example, the individual functions are weighted and added together. Since the weighted sum method is limited to only calculate non-dominated points that lie on the convex hull of the Pareto front, we also consider another approach with the hypervolume scalarization.[5] The hypervolume indicator function is defined by the volume spanned by the computed outcome vectors (of feasible solutions) and an appropriately chosen reference point, and is aimed to be maximized.

The goal of this thesis is to develop scalarization-based optimization methods to PDE constrained shape optimization problems in order to approximate the Pareto front. We address two distinct biobjective shape optimization problems. Firstly, we consider ceramic components under tensile load. We want to maximize the mechanical stability while minimizing the volume. The mechanical stability is modeled by a Weibull type formulation of the probability of failure. Secondly, we consider turbomachinery components in order to maximize aerodynamic efficiency and mechanical reliability. Thereby, we use the state-of-the-art adjoint fluid dynamics solver TRACE of the German Aerospace Center to evaluate the aerodynamic efficiency.[6] The mechanical reliability is modeled by a low cycle fatigue objective. With this choice of models we benefit from having gradient information available for all objectives, enabling the utilization of gradient based optimization methods.

GIVEN research project Energy supply is a challenging task nowadays. In the conventional way energy production is based on fossil resources. However, since fossil fuels, like coal or natural gas, are limited, the expansion of renewable energies, such as solar energy or wind power, is necessary. In the German government's plans for the energy turnaround, the share of renewable energy in the power grid is expected to rise

[4] See Miettinen and Mäkelä, 2002, for example.
[5] See Zitzler and Thiele, 1999.
[6] *TRACE User Guide* 2019.

to over 60% in 2050.[7] However, the main disadvantage of these environmentally friendly energies is that they are subject to volatility. With increasing share of renewable energies, the amount of volatile energy supply increases. Thus, energy networks need back up solutions to cover the energy demand in case of insufficient electricity production from renewable energies. Gas turbines or the combination with steam turbines are an appropriate solution for flexible reserve power plants, mainly because of short start-up times. They can also be important as a bridge technology for CO_2 reduction.[8] Among fossil fuels, gas has the highest efficiency with over 63%.[9] In the future a 100% renewable energy supply may be targeted after the year 2050. Also in this scenario high efficient gas turbines are considerable if hydrogen and methane are used as storage media from renewable energies and can be converted into electricity with gas turbines if required.[10]

In the design process of gas turbines certain design requirements must be taken into account. On the one hand the efficiency of energy conversion is aimed to be optimized and on the other hand, it is also important to ensure reliability and flexibility, especially with regard to low-cycle fatigue. Low-cycle fatigue (LCF) is the gradual structural deterioration of a material subjected to cyclic loading, usually requiring only a relatively small number of load cycles to failure.[11] This phenomenon is particularly relevant in the design of gas turbines due to the frequent and large variations in operating conditions such as start-up, shutdown and load cycling. By considering aspects of low-cycle fatigue in addition to energy conversion efficiency, gas turbine designers can ensure not only optimum performance but also long-term reliability and operational flexibility, improving the overall efficiency of the turbine system.

The project *"Shape Optimization of Gas Turbines in Volatile Energy Networks"*, abbreviated by *GivEn*, is a research project, where the extensive design process of gas turbines is considered. The aim is to develop mathematical methods and implement a multiobjective optimization process to evaluate different designs taking into account the objectives fluid-dynamic efficiency as well as mechanical reliability.[12]

---

[7] *The Energy of the Future. 8th Monitoring Report on the Energy Transition –Reporting Years 2018 and 2019* 2021.

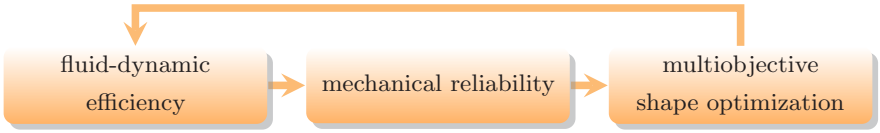
[8] *The Energy of the Future. 8th Monitoring Report on the Energy Transition –Reporting Years 2018 and 2019* 2021.

[9] *GE's HA Gas Turbine Delivers Second World Record for Efficiency* 2018.

[10] Sachverständigenrat für Umweltfragen, 2011.

[11] Gottschalk, Saadi, et al., 2018.

[12] `https://www.given-project.de/`



This project was funded by the Federal Ministry of Education and Research (BMBF), grant-no: 05M18PXA. This support is gratefully acknowledged.

Outline of this thesis The thesis is structured into two main parts.

In the first part, outlined in Chapters 2 to 3, we consider general multiobjective optimization problems and concentrate on two scalarization techniques. In Chapter 2 the concept of multiobjective optimization and the well-known weighted sum method are explained, while Chapter 3 introduces a novel hypervolume scalarization technique.

The second part commences in Chapter 4 and focuses on the application of these methods to two specific PDE constrained shape optimization problems.

In Chapter 5 we consider ceramic components under tensile load, where we want to minimize cost and the probability of failure. For this biobjective shape optimization problem we start by implementing the weighted sum method and utilize B-spline parameterization for the shape representation. Subsequently, we explore the same problem using free form deformation and implement the hypervolume scalarization technique.

In Chapter 6 we focus on gas turbine blades and consider the two optimization goals aerodynamic efficiency and mechanical reliability. We set up a coupling routine using several elements: the TRACE software package to asses the efficiency, an LCF solver to evaluate the reliability and the TRASOR software package for calculating Steklov-Poincaré gradients. Subsequently, the combination of these elements with the integration of the weighted sum method is described and applied to the gas turbine profile of the T106A configuration.

This research is based on the research project *GivEn* and parts of this thesis are already published. We refer to the respective publications in the corresponding chapters of this thesis.

Related literature Before delving into our own research findings, we briefly review some related literature in the context of (multiobjective) shape optimization.

Historically, shape optimization problems focus on single objective optimization problems. For a general introduction to shape optimization, we recommend Allaire, 2002; Bucur and Buttazzo, 2005; Haslinger and Mäkinen, 2003; Sokolowski and Zolesio,

1992. Often, mechanical integrity is considered as objective function to assess a shape subjected to loads, see, for example, Allaire and Jouve, 2008; Duysinx and Bendsøe, 1998; Haslinger and Mäkinen, 2003; Picelli et al., 2018. For material science we refer to Rösler, Harders, and Bäker, 2019. However, most models for the mechanical integrity are based on min-max formulations and are hence not differentiable at the points of highest vulnerability. This can be overcome by assessing the probability of failure using Poisson point processes as in Brückner-Foit et al., 1997; Munz and Fett, 2001; Roudi, Riesch-Oppermann, and Kraft, 2005, an approach based on a probabilistic description for the ultimate strength of ceramic material, see Weibull, 1939, who laid the foundation to the reliability objective that is also used here, see Bolten, Gottschalk, Hahn, et al., 2019; Bolten, Gottschalk, and S. Schmitz, 2015; Gottschalk and Saadi, 2019; Gottschalk, Saadi, et al., 2018; Gottschalk and S. Schmitz, 2014; S. Schmitz, 2014; S. Schmitz, Rollmann, et al., 2013; S. Schmitz, Seibel, et al., 2013.

Multiobjective formulations of shape optimization problems are not so widely spread. They are partly mentioned in Haslinger and Mäkinen, 2003. The existence of Pareto optimal shapes are, for example, considered in Allaire, 2002; Chirkov et al., 2018; Fujii, 1988; Haslinger and Mäkinen, 2003. Further research contributions that suggest meta heuristic approaches are given, among others, in Chirkov et al., 2018; Deb, 2001; K. Deb, 2011; K. Deb and Goel, 2002; Zavala et al., 2013. Evolutionary algorithms and gradient based approaches are compared in Pulliam et al., 2003.

Overall, the field of multiobjective shape optimization is a dynamic area in which ongoing research is addressing various challenges. Further literature is mentioned in the corresponding chapters.

Chapter 2

Multiobjective Optimization

In this chapter we provide a brief introduction to multiobjective optimization based on Ehrgott, 2005 and set fundamentals for the following chapters. For a thorough introduction to multiobjective optimization we refer to Ehrgott, 2005; Miettinen, 1998. We start in Section 2.1 with a general formulation of multiobjective optimization problems. In Section 2.2 we give an insight in selected solution approaches for multiobjective optimization problems and refer to related literature. We explain the weighted sum method in Section 2.2.1 as an example for scalarization techniques and also address some drawbacks of the method, which were already analyzed in Das and Dennis, 1997. In Section 2.2.2 we explain the steepest descent algorithm, see also, for example, Geiger and Kanzow, 1999, for solving single-objective optimization problems such as the mentioned weighted sum scalarizations. In Section 2.2.3 we conclude this chapter with a short review of the multiobjective descent algorithm by Fliege and Svaiter, 2000, since we will compare this algorithm to the scalarization methods used in this thesis.

2.1 General Definitions

A *multiobjective optimization problem* (MOP) is given by

$$\begin{aligned} \min \quad & f(x) = (f_1(x), \ldots, f_p(x))^\top \\ \text{s. t.} \quad & x \in \mathcal{X}. \end{aligned} \tag{MOP}$$

Here, $\mathcal{X}$ is the set of feasible solutions and $f = (f_1, \ldots, f_p) \colon \mathcal{X} \to \mathbb{R}^p$ is the vector-valued objective function comprising p individual objective functions $f_k \colon \mathcal{X} \to \mathbb{R}$ with $k = 1, \ldots, p$. We assume that $\mathcal{X}$ is nonempty and hence also $f(\mathcal{X}) \subset \mathbb{R}^p$ is nonempty. We call the objective vector $f(x)$ of a feasible solution $x \in \mathcal{X}$ feasible *point* or feasible *outcome vector*. In the cases $p = 2$ and $p = 3$ we call the (MOP) a *biobjective* and *triobjective* optimization problem, respectively.

For an exemplary biobjective optimization problem in Figure 2.2b the ideal point, which is given by the minimal objective value of f_1 and f_2, is shown. In general, the *ideal point* $y^{\mathrm{I}} = \left(y_1^{\mathrm{I}}, \ldots, y_p^{\mathrm{I}}\right)^\top$ of (MOP) is defined by

$$y_k^{\mathrm{I}} = \min_{x \in \mathcal{X}} f_k(x), \quad k = 1, \ldots, p. \tag{2.1}$$

Since the ideal point is usually not included in the set of feasible outcome vectors, i.e., $y^{\mathrm{I}} \notin f(\mathcal{X})$, as we have seen in Figure 2.2b, we need another concept of "optimality". According to the *Pareto concept of optimality* we say that a feasible solution is Pareto optimal if no individual objective can be improved without deteriorating one of the others. This concept is based on the *component-wise order*, see Ehrgott, 2005, which we define in the following. For $y, y' \in \mathbb{R}^p$, we have:

$$\begin{aligned}
y < y' \quad &\iff \quad y_k < y'_k \text{ for } k = 1, \ldots, p \quad \text{(strict component-wise order)} \\
y \leqq y' \quad &\iff \quad y_k \leq y'_k \text{ for } k = 1, \ldots, p \quad \text{(weak component-wise order)} \\
y \leqslant y' \quad &\iff \quad y \leqq y' \text{ and } y \neq y' \qquad\qquad \text{(component-wise order)}
\end{aligned}$$

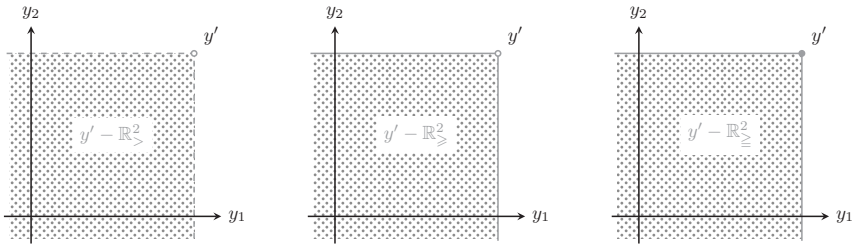
We say y *dominates* y' if it holds that $y \leqslant y'$ for $y, y' \in \mathbb{R}^p$. Additionally, we can also say y dominates y' w.r.t. the weak component-wise order and strict component-wise order if $y \leqq y'$ and $y < y'$ for $y, y' \in \mathbb{R}^p$, respectively. The converse relations $\geq, \geqslant$ and $\geqq$ are defined analogously. Accordingly, the sets $\mathbb{R}^p_>, \mathbb{R}^p_\geqslant$ and $\mathbb{R}^p_\geqq$ are defined as follows:

$$\mathbb{R}^p_> \coloneqq \{y \in \mathbb{R}^p \mid y_k > 0, k = 1, \ldots, p\} \tag{2.2}$$

$$\mathbb{R}^p_\geqslant \coloneqq \{y \in \mathbb{R}^p \mid y_k \geq 0, k = 1, \ldots, p, y \neq 0\} \tag{2.3}$$

$$\mathbb{R}^p_\geqq \coloneqq \{y \in \mathbb{R}^p \mid y_k \geq 0, k = 1, \ldots, p\} \tag{2.4}$$

Figure 2.1 illustrates the component-wise order for $p = 2$ by showing, for a given outcome vector $y' \in \mathbb{R}^2$, the sets $y' - \mathbb{R}^2_>$, $y' - \mathbb{R}^2_\geqslant$ and $y' - \mathbb{R}^2_\geqq$, that contain all points that would potentially dominate y' w.r.t. the strict component-wise, the weak



(a) $y < y'$ (b) $y \leqslant y'$ (c) $y \leqq y'$

Figure 2.1: Component-wise order for $p = 2$.

component-wise and the component-wise order, respectively.

Now we can formally define a Pareto optimal solution.

Definition 1 (Pareto optimality)**.** *A feasible solution $x \in \mathcal{X}$ of* (MOP) *is called* Pareto optimal *or* efficient *if there is no $\hat{x} \in \mathcal{X}$ such that $f(\hat{x}) \leqslant f(x)$, i.e., $\nexists\, \hat{x} \in \mathcal{X}$ s. t. $f_k(\hat{x}) \leq f_k(x)\, \forall k \in \{1, \ldots, p\}$ and $f(\hat{x}) \neq f(x)$.*

Moreover, a feasible solution x is called weakly Pareto optimal *or* weakly efficient *if there is no $\hat{x} \in \mathcal{X}$ such that $f(\hat{x}) < f(x)$, and a feasible solution x is called* strictly efficient*, is there is no $\hat{x} \in \mathcal{X}$, $\hat{x} \neq x$ such that $f(\hat{x}) \leqq f(x)$.*

The corresponding outcome vector $f(x)$ of a (weakly) Pareto optimal solution is called (weakly) non-dominated *point or (weakly)* non-dominated *outcome vector.*

We denote the set of efficient solutions by $\mathcal{X}_{\mathrm{E}}$ and call it *efficient set*. The set of non-dominated outcome vectors $\mathcal{Y}_{\mathrm{N}} \coloneqq \{f(x) \,|\, x \in \mathcal{X}_{\mathrm{E}}\}$ is called *Pareto front* or *non-dominated set*. For the corresponding sets of weakly and strictly efficient solutions we write $\mathcal{X}_{\mathrm{wE}}$ and $\mathcal{X}_{\mathrm{sE}}$, respectively. In addition, let $\mathcal{Y}_{\mathrm{wN}} \coloneqq \{f(x) \,|\, x \in \mathcal{X}_{\mathrm{wE}}\}$ denote the set of weakly non-dominated outcome vectors.

Apparently, adding the negative orthant $-\mathbb{R}^p_{\geqslant}$ to a non-dominated point $y \in \mathcal{Y}_{\mathrm{N}}$ results by definition in an empty intersection with $f(\mathcal{X})$, i.e., $f(\mathcal{X}) \cap \left(y - \mathbb{R}^p_{\geqslant}\right) = \emptyset$. Otherwise, the point y would not be non-dominated in the first place. Thus, a point $y \in \mathbb{R}^p$ is non-dominated if and only if $f(\mathcal{X}) \cap \left(y - \mathbb{R}^p_{\leqq}\right) = \{y\}$.

Figure 2.2a shows an example of an objective space with two objectives. The Pareto front $\mathcal{Y}_{\mathrm{N}}$ is highlighted in blue and two points $f(x'), f(x'') \in \mathcal{Y}_{\mathrm{N}}$ are marked with the negative orthant, which is represented by a dotted pattern.

In Figure 2.2b we consider the same objective space. As already mentioned, the ideal point is drawn there. In addition, the *Nadir point* y^{N} is marked, which is composed of the worst individual objective values over the efficient set. Furthermore,

the point $\bar{y}$ is simply given by the worst individual objective values over the entire feasible set and can be considered as a global upper bound.

Thus, the Nadir point $y^{\mathrm{N}} = \left(y_1^{\mathrm{N}}, \ldots, y_p^{\mathrm{N}}\right)^\top$ is given by

$$y_k^{\mathrm{N}} = \max_{x \in \mathcal{X}_{\mathrm{E}}} f_k(x), \quad k = 1, \ldots, p \tag{2.5}$$

and a global upper bound $y^{\mathrm{UB}} = \left(y_1^{\mathrm{UB}}, \ldots, y_p^{\mathrm{UB}}\right)^\top$ is given by

$$y_k^{\mathrm{UB}} = \max_{x \in \mathcal{X}} f_k(x), \quad k = 1, \ldots, p. \tag{2.6}$$

Note that the ideal point, Nadir point and the global upper bound defined in (2.1), (2.5) and (2.6), respectively, do not necessarily exists if, for example, $f(\mathcal{X})$ is an open set or unbounded. Although we have assumed that $\mathcal{X}$ and $f(\mathcal{X})$ are nonempty, it may also be the case that $\mathcal{X}_{\mathrm{E}}$ is empty. Unless otherwise stated, we assume from now that $\mathcal{X}_{\mathrm{E}}$ is nonempty.

Furthermore, we provide the definitions of properly efficient solutions and supported efficient solutions.

Definition 2 (Proper efficiency (Geoffrion, 1968))**.** *An efficient solution $\hat{x}$ is called* properly efficient *in Geoffrion's sense if there exists as scalar $M > 0$ such that for all $x \in \mathcal{X}$ and for all $k \in \{1, \ldots, p\}$ with $f_k(x) < f_k(\hat{x})$, it holds that*

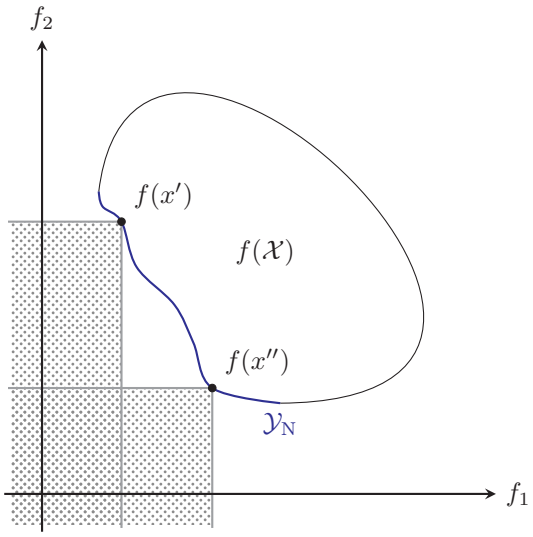
$$\frac{f_k(\hat{x}) - f_k(x)}{f_i(x) - f_i(\hat{x})} \leq M \tag{2.7}$$

for some $i \in \{1, \ldots, p\}$ with $f_i(x) > f_i(\hat{x})$. The corresponding outcome vector $f(\hat{x})$ is called properly non-dominated*.*

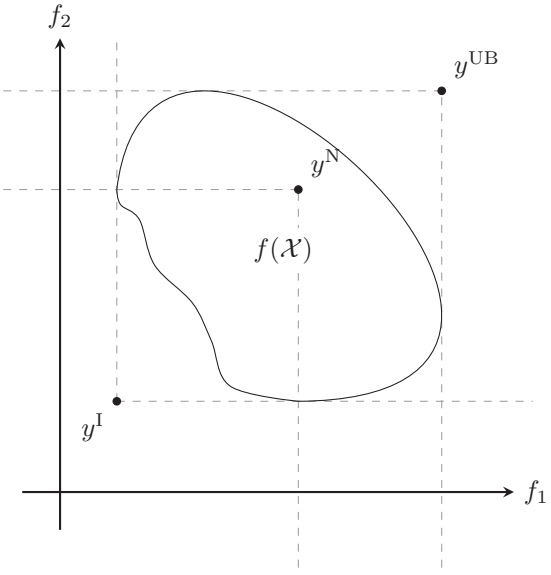
We denote the set of all properly efficient solutions by $\mathcal{X}_{\mathrm{pE}}$ and the set of all properly non-dominated outcome vectors by $\mathcal{Y}_{\mathrm{pN}}$. In other words, $\mathcal{Y}_{\mathrm{pN}}$ is the set of all non-dominated points with bounded trade-offs. Figure 2.3 shows an example where $\mathcal{Y}_{\mathrm{pN}} \subsetneqq \mathcal{Y}_{\mathrm{N}}$, because the points y^1 and y^2 are non-dominated but not properly non-dominated, see also Ehrgott, 2005 for a similar example.

A further definition is given for supported efficient solution. This definition is important for the weighted sum method in Section 2.2.1 since the weighted sum method can only find supported efficient solution.

Definition 3 (Supported efficiency)**.** *An efficient solution $\hat{x} \in \mathcal{X}_E$ is called* supported efficient *solution if there is no point $y \in conv\{f(x) \,|\, x \in \mathcal{X}\}$ dominating $f(\hat{x})$.*

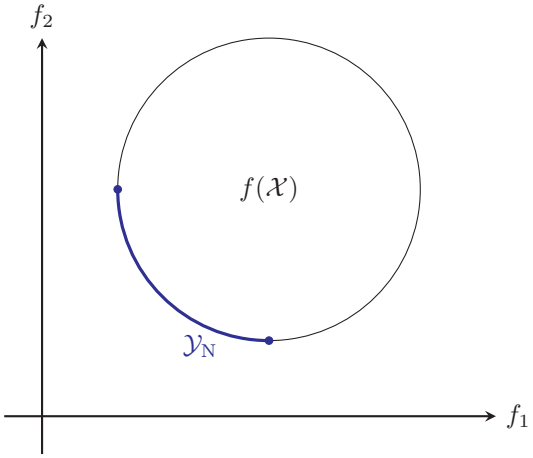


(a) Pareto front.

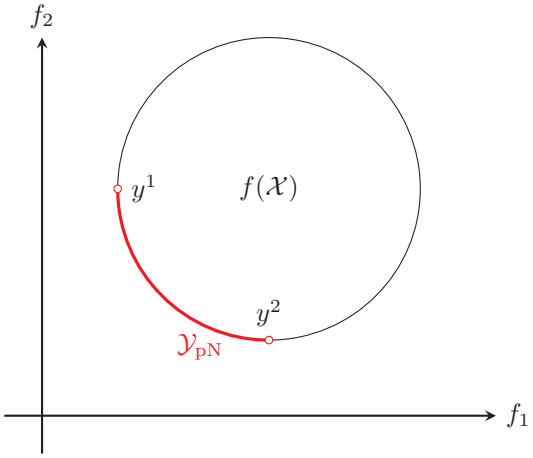


(b) Ideal point y^{I}, Nadir point y^{N} and upper bound y^{UB}.

Figure 2.2: Example for a feasible set of outcome vectors for a biobjective optimization problem.



(a) Non-dominated points.



(b) Properly non-dominated points.

Figure 2.3: Example for properly non-dominated points.

In Figure 2.2a it can be seen that the solution x'' is a supported efficient solution because $f(x'')$ lies on the boundary of $\text{conv}\{f(\mathcal{X})\}$. However, the solution x' is not a supported efficient solution because $\left(f(x') - \mathbb{R}^p_{\geqslant}\right) \cap \text{conv}\{f(\mathcal{X})\} \neq \emptyset$.

2.2 Solution Approaches and Related Literature

In general, there is more than one Pareto optimal solution of (MOP) and it is usually difficult to obtain a complete representation of the Pareto front. Therefore, often only a set of Pareto optimal solutions is computed to represent the Pareto front. When searching for solutions to problem (MOP), we distinguish between (potentially) exact methods and heuristic methods. We first have a look at (potentially) exact methods.

A common approach to finding solutions of a multiobjective optimization problems (MOP) are scalarization techniques where the original problem (MOP) is replaced by a series of related single-objective optimization problems and then solved by using appropriate single-objective methods. If associated single-objective problems are solved to global optimality, then exact Pareto optimal solutions are obtained. The weighted sum method, as an example for a scalarization technique, is a simple approach where the multidimensional objective function is replaced by a weighted sum of the individual objective functions without changing the feasible set. Thus, by solving the weighted sum problem for various selection of different non-negative weights, a set of solutions for (MOP) may be computed. The weighted sum method is explained in Section 2.2.1. Other examples for scalarization techniques are the ε-constrained method, the hybrid method and many more. There is a whole range of different scalarization methods, see, e.g., Ehrgott, 2005; Miettinen and Mäkelä, 2002. The hypervolume method, which we present in Chapter 3, is also a scalarization technique.

In contrast to scalarization techniques, there also exist multiobjective versions of gradient descent methods or Newton methods. These methods also belong to the (potentially) exact methods. Especially worth mentioning here is the multiobjective descent algorithm by Désidéri, 2009; Fliege and Svaiter, 2000 which we briefly explain in Section 2.2.3.

Furthermore, there are also evolutionary multiobjective optimization (EMO) algorithms, see Deb, 2001. These are heuristic methods that aim at an efficient and high quality approximation of points in $\mathcal{Y}_{\text{N}}$. These methods are population based metaheuristic algorithms inspired by the biological evolution. Starting with an initial population of solutions, the "fitness" of the individuals will be evaluated and then

candidates for creating new solutions will be selected. For related literature, see, e.g., K. Deb, 2011; K. Deb and Goel, 2002.

In this thesis we will on focus two different scalarization techniques, the weighted sum method introduced in Section 2.2.1 and the hypervolume method introduced in Chapter 3. We do this because we consider from Chapter 4 PDE constrained shape optimization problems with more than one objective where the function evaluations are expensive and additional constraints are not easy to handle. Therefore, we want to use methods, that do not change the feasible set or where the change in the feasible set can be ignored in the numerical implementation, under certain circumstances. In many other methods often further restrictions are introduced, such as in the ε-constrained method, for example, or the solutions of additional sub-problems are required.

2.2.1 Weighted Sum Method

The *weighted sum scalarization* of a multiobjective optimization problem (MOP) is given by

$$\begin{aligned} \min \quad & \omega^\top f(x) = \sum_{k=1}^{p} \omega_k f_k(x) \\ \text{s. t.} \quad & x \in \mathcal{X} \end{aligned} \tag{WS-SP}$$

with the weighting vector $\omega = (\omega_1, \ldots, \omega_p)^\top$ such that $\omega \geqslant 0$, i.e., $\omega \neq 0$ and $\omega_k \geq 0$ for all $k \in \{1, \ldots, p\}$.

Since we aim at minimizing all individual objective functions f_k, we also aim at minimizing the weighted sum of them. By varying the weights, we want to calculate an approximation of the Pareto front. The weights can be interpreted as preferences, i.e., the larger the weight ω_k is chosen in relation to the other weights, the more important is it to minimize the individual function f_k.

Furthermore, in the case of having an individual objective function $f_{k'}$, that has to be maximized, we also can use a negative weight $\omega_{k'} \leq 0$ instead of replacing $f_{k'}$ by $-f_{k'}$. Nevertheless, if not stated otherwise, we always assume that all individual objective functions are given in minimization form as in (MOP). Without loss of generality ω can be re-scaled such that $\omega_k \in [0, 1]$ and $\sum_{k=1}^{p} \omega_k = 1$.

In the following we discuss some characteristics of the weighted sum method and also name some drawbacks of this method. Figure 2.4 illustrates the concept of the weighted sum method. For a level $c \in \mathbb{R}$ the level set of the weighted sum scalarization

is given by

$$\mathcal{L}^W_{=}(c;\omega) \coloneqq \left\{ y \in \mathbb{R}^p \,\middle|\, \omega^\top y = c \right\}. \tag{2.8}$$

Obviously, the level sets are lines in the objective space, where the slope is defined by the weighting vector ω. Thus, in order to minimize the weighted sum objective value, the line corresponding to the level $c = c_1$ in Figure 2.4 will be shifted as far as possible to the lower left, i.e., the value c will be minimized as far as possible, while the line intersects the feasible set $f(\mathcal{X})$. At level $c = c_2$, for example, we already have intersection with $\mathcal{Y}_\mathrm{N}$, i.e., the intersection points with $\mathcal{Y}_\mathrm{N}$ are a non-dominated, but they are not optimal for the chosen weighted sum scalarization, because the level c can be further minimized. The minimal value for c is obtained at c^*. This value is then the objective value of an optimal solution of (WS-SP) and the resulting intersection point $f(x'')$ is a non-dominated point of (MOP).

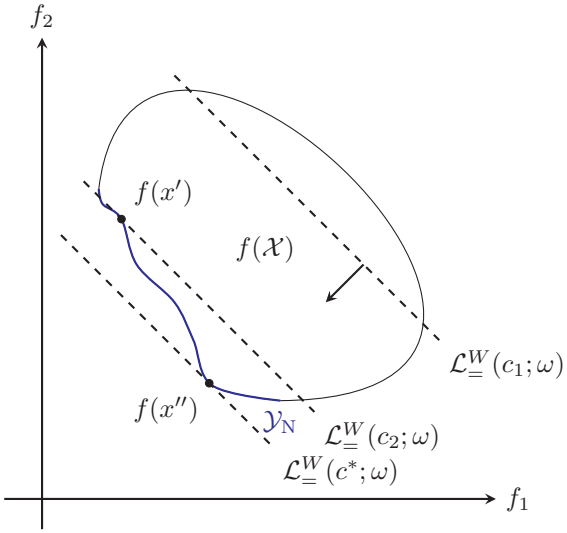


Figure 2.4: Concept of the weighted sum method.

It can be shown that if we have an optimal solution of the weighted sum scalarization with positive weights, we have an efficient solution of the multiobjective counterpart. More formally, we have the following theorem:

Theorem 4. *If x^* is an optimal solution of the weighted sum scalarization* (WS-SP)

of (MOP) *with weighting vector* $\omega \geq 0$*, then the following statements holds:*

(i) If $\omega \geq 0$*, then* x^* *is a weakly efficient solution of* (MOP)*.*

(ii) If $\omega > 0$*, then* x^* *is an efficient solution of* (MOP)*.*

(iii) If $\omega \geq 0$ *and* x^* *the unique solution of* (WS-SP)*, then* x^* *is a strictly efficient solution of* (MOP)*.*

The proof can be found, e.g., in Ehrgott, 2005. Conversely, we do not have in general that for every efficient solution x^* there exists a corresponding weighting vector, such that x^* is an optimal solution of the corresponding weighted sum scalarization. The reason for this is that non-convex parts of the Pareto front cannot be recovered by weighted sum scalarizations. An example for this can be seen in Figure 2.4, where no weighting vector exits such that the point $f(x')$ can be obtained by the weighted sum method.

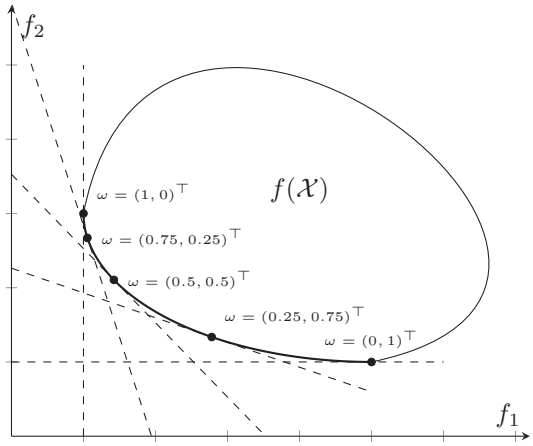


Figure 2.5: Even distribution of weighting vectors.

Furthermore, we have to be careful when choosing the weighting vectors when aiming at an approximation of the Pareto front. In Figure 2.5 the objective space of a biobjective optimization problem is shown. Here, the set of feasible outcome vectors is convex. The solutions of the weighted sum method are highlighted for $\omega = (\lambda, 1 - \lambda)$ with $\lambda = 0, 0.25, 0.5, 0.75, 1$. Obviously, an even distribution of weights does not necessarily lead to an even distribution of solutions. We refer to Das and Dennis, 1997 for a detailed analysis of this problem. Especially, if the individual objectives differ in the scale of objective values, the weights have to be chosen with case.

2.2.2 Single-Objective Steepest Descent Algorithm

In Chapter 5 and 6 we will apply scalarization techniques to two different shape optimization problems. Since gradient information is available in our applications, we can use the steepest descent algorithm for the solution of the scalarized problems such as the weighted sum scalarization. In the following, we briefly discuss the idea of descent methods based on Geiger and Kanzow, 1999.

The steepest descent algorithm is a method to find stationary points of a single-objective function. A *stationary point* of a differentiable function $f_s : \mathbb{R}^n \to \mathbb{R}$ is a point where the gradient is equal to zero, i.e., $x \in \mathbb{R}^n$ is stationary point of f_s if $\nabla f_s(x) = 0$. We consider unconstrained single-objective optimization problems:

$$\begin{aligned} \min \quad & f_s(x) \\ \text{s. t.} \quad & x \in \mathbb{R}^n \end{aligned} \tag{SP}$$

where $f_s \colon \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function. Thus, the objective function f_s here is a scalar-valued function.

An iterative line search scheme can generally be described as in Algorithm 1.

Data: Choose starting solution $x^{(0)} \in \mathbb{R}^n$, set $i := 0$.
Output: Solution $x^{(i)}$.
while *no stopping condition for* $x^{(i)}$ *is fulfilled* **do**
  Determine a search direction $d^{(i)}$;
  Calculate a step length $t_i > 0$;
  Update $x^{(i)} = x^{(i)} + t_i d^{(i)}$, set $i = i + 1$;
end

Algorithm 1: Iterative Line Search Scheme

Since we want to minimize, a descent direction must be chosen as the search direction. We call $d \in \mathbb{R}^n$ a *descent direction* at the point $x \in \mathbb{R}^n$ if there exits a step length $\bar{t} > 0$ such that $f_s(x + td) < f_s(x)$ for all $t \in (0, \bar{t}\,]$. A sufficient condition for a descent direction is given by the following lemma (for the proof see Geiger and Kanzow, 1999).

Lemma 5. *Let $f_s \colon \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function, $x \in \mathbb{R}^n$ and $d \in \mathbb{R}^n$. If $\nabla f_s(x)^\top d < 0$, then d is a descent direction of f_s at the point x.*

Using the negative gradient at a non-stationary point as a search direction, i.e.,

$d = -\nabla f_s(x)$, we have with

$$\nabla f_s(x)^\top d = -\|\nabla f_s(x)\| < 0 \tag{2.9}$$

a descent direction. The method is then called *gradient descent algorithm* or *steepest descent algorithm*.

Later in this thesis we implement the steepest descent algorithm. For calculating a step length we use the Armijo rule: First we choose $\sigma \in (0,1)$ and $\beta \in (0,1)$. Then we determine $t_i := \max\left\{\beta^\ell \,\middle|\, \ell = 0,1,2,\dots\right\}$ for the current iterate $x^{(i)}$ and search direction $d^{(i)}$ such that

$$f(x^{(i)} + t_i d^{(i)}) \leq f_s(x) + \sigma t_i \nabla f_s(x^{(i)})^\top d^{(i)}. \tag{2.10}$$

The stopping condition for $x^{(i)}$ is often chosen as $\|\nabla f_s(x^{(i)})\| < \varepsilon$ with $\varepsilon > 0$ sufficiently small. Moreover, we additionally limit the number of iterations, i.e., $i \leq i_{\max}$ for a previously chosen $i_{\max} \in \mathbb{N}$.

For a detailed introduction to solutions approach for unconstrained optimization problem see, e.g., Geiger and Kanzow, 1999.

2.2.3 Multiobjective Descent Algorithm

In this section we briefly explain the multiobjective descent algorithm. Later in this thesis, we want compare the multiobjective descent algorithm to the hypervolume method presented in Chapter 3. Additionally, part of this thesis is the implementation of weighted sum method for a shape optimization problem introduced in Chapter 5. The results were already published in Doganay et al., 2020 including a comparison between the numerical solutions obtained by the weighted sum method and the numerical solutions obtained by the multiobjective descent algorithm. This will be discussed in Section 5.4. The explanation is based on Doganay et al., 2020.

The multiobjective descent algorithm can be understood as a natural generalization of single-objective gradient descent algorithms and was proposed in Fliege and Svaiter, 2000, see also Fliege, Vaz, and Vicente, 2019. Similar approaches have been suggested in Désidéri, 2009, 2012; Giacomini, Désidéri, and Duvigneau, 2014

First we have to state the definition of Pareto critical solutions. If derivative information is available, necessary optimality conditions can be formulated that generalize the concept of critical points from single-objective optimization. In the following, we assume that the (MOP) is unrestricted, thus the feasible set is given

by $\mathcal{X} = \mathbb{R}^n$. Assuming that all individual objective functions are continuously differentiable, a necessary condition for a solution $x \in \mathbb{R}^n$ to be locally Pareto optimal is that

$$\left\{ d \in \mathbb{R}^n \ : \ \nabla f_k(x)^\top d < 0, \ k = 1, \ldots, p \right\} = \emptyset, \tag{2.11}$$

i.e., there does not exist a direction $d \in \mathbb{R}^n$ that is a descent direction for all individual objectives. If $x^* \in \mathbb{R}^n$ satisfies this condition we call it a *Pareto critical solution*.

Now, the multiobjective descent algorithm is an iterative process starting with a solution $x^{(0)}$, where in every iteration all individual objective functions will improved simultaneously, i. e., $f\left(x^{(i+1)}\right) < f\left(x^{(i)}\right)$. This is based on the observation that if a solution $x \in \mathbb{R}^n$ is not Pareto critical according to (2.11), then there exists a direction $d \in \mathbb{R}^n$ which is a descent direction all single objectives. Thus, if in an iterative solution method the current iterate $x^{(i)} \in \mathbb{R}^n$ is not Pareto critical, a *direction of steepest biobjective descent* $d^{(i)} \in \mathbb{R}^n$ can be defined according to Fliege and Svaiter, 2000 as a direction solving the auxiliary optimization problem

$$\begin{aligned} \min_{\rho \in \mathbb{R}, d \in \mathbb{R}^n} \quad & \rho + \frac{1}{2}\|d\|^2 \\ \text{s.t.} \quad & \nabla f_k \left(x^{(i)}\right)^\top d \leq \rho, \ k = 1, \ldots, p. \end{aligned} \tag{2.12}$$

Problem (2.12) is a convex quadratic optimization problem with linear inequality constraints. Note that the term $\frac{1}{2}\|d\|^2$ in the objective function ensures that the problem is bounded and that the solution $\rho = 0$, $d = 0$ is always feasible. Note also that the optimal value ρ^* is negative if and only if $d^* \neq 0$, i. e., if a direction of steepest multiobjective descent exists.

When a direction of steepest multiobjective descent $d^{(i)} \neq 0$ is found, then we move from $x^{(i)}$ into the direction $d^{(i)}$ to a new point $x^{(i+1)} := x^{(i)} + t_i d^{(i)}$. The step length $t_i > 0$ is computed using an Armijo-like rule. A step length t is accepted if it guarantees a sufficient multiobjective descent in the sense that

$$f_k \left(x^{(i)} + t\, d^{(i)}\right) \leq f_k \left(x^{(i)}\right) + \beta\, t\, \nabla f_k \left(x^{(i)}\right)^\top d^{(i)}, \ k = 1, \ldots, p \tag{2.13}$$

with $\beta \in (0, 1)$ be a predefined constant. A proof for the finiteness of this procedure is given in Fliege and Svaiter, 2000.

The overall method is summarized in Algorithm 4.

Data: Choose starting solution $x^{(0)} \in \mathbb{R}^n$ and $\beta \in (0,1)$, set $i := 1$.
Result: Solution $x^{(i)}$.
while *no stopping condition for* $x^{(i)}$ *is fulfilled* **do**
  Compute $d^{(i)}$ as a solution of (2.12);
  Compute a step length $t_i > 0$ according to (2.13);
  Update $x^{(i+1)} := x^{(i)} + t_i\, d^{(i)}$ and $i = i + 1$;
end

Algorithm 2: Multiobjective descent algorithm according to Fliege and Svaiter, 2000.

Chapter 3

Hypervolume Scalarization

In the following section we will introduce the hypervolume scalarization. The results in this section have been published in Schultes et al., 2021 for the biobjective case. Here, we present a generalization to multiobjective problems with an arbitrary number of objectives, i.e., for arbitrary $p \geq 2$. The hypervolume scalarization is based on the hypervolume indicator that was introduced by Zitzler and Thiele, 1999 as a measure to compare the performance of evolutionary multiobjective algorithms. For more related literature see Section 3.1.

For a multiobjective optimization problem (MOP) the hypervolume indicator measures the hypervolume spanned by an outcome vector $f(x)$ (or a set of outcome vectors $f(S) \coloneqq \{f(x) \mid x \in S \subset \mathcal{X}\}$) and a predefined reference point $r \in \mathbb{R}^p$, i.e., for a set of solutions $S \subset \mathcal{X}$ and a reference point $r \in \mathbb{R}^p$ the *hypervolume indicator* is given by

$$H(f(S)) = \operatorname{Vol}\left(\bigcup_{x \in S} \left(f(x) + \mathbb{R}^p_{\leqq}\right) \cap \left(r - \mathbb{R}^p_{\leqq}\right)\right), \tag{3.1}$$

with $\mathbb{R}^p_{\leqq}$ defining the positive orthant, see (2.4). For a single solution $x \in \mathcal{X}$, i.e., $|S| = 1$, the hypervolume indicator simplifies to

$$H(f(x)) = \prod_{k=1}^{p} \left(r_k - f_k\left(x\right)\right). \tag{3.2}$$

While in EMO algorithms the hypervolume indicator is mainly used to evaluate the quality of already computed solutions (or solution sets, respectively), it can also be used as a scalarization method for (MOP). For a given reference point

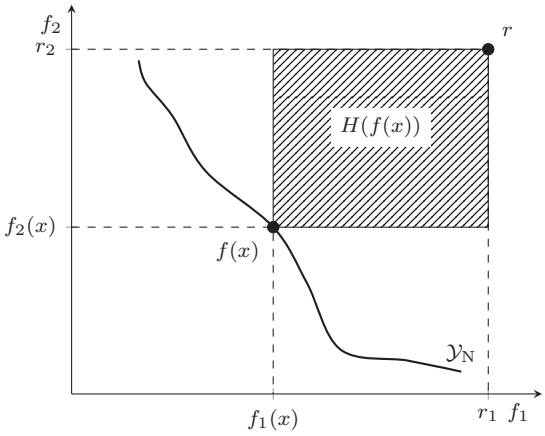


Figure 3.1: Hypervolume contribution $H(f(x))$ of a solution x with respect to the reference point r (cf. Schultes et al., 2021).

$r \coloneqq (r_1, \ldots, r_p)^\top \in \mathbb{R}^p$ the *hypervolume scalarization* of (MOP) is formulated by

$$\begin{aligned} \max \quad & H(f(x)) = \prod_{k=1}^{p} (r_k - f_k(x)) \\ \text{s. t.} \quad & f(x) \leqq r, \\ & x \in \mathcal{X}. \end{aligned} \tag{HV-SP}$$

We thus want to find a solution $x \in \mathcal{X}$ such that the dominated hypervolume w.r.t. $f(x)$ is maximized, i.e., we aim at maximizing the volume spanned by the predefined reference point and the possible outcome vectors in order to find an efficient solution of (MOP).

In the following, we assume that the reference point r is chosen such that the set $\{x \in \mathcal{X} \mid f(x) \leqq r\} \neq \emptyset$. Additionally, we assume that an optimal solution of (HV-SP) exists.

The constraint $f(x) \leqq r$ is essential since also points that are dominated by the reference point in an even number of objectives have positive hypervolume values and are thus potential candidates for maximizing $H(f(x))$. Thus, with the constraint $f(x) \leqq r$ we ensure that $H(f(x))$ is non-negative and that feasible points of (HV-SP) are not dominated by the reference point r.

However, the hypervolume scalarization only guarantees efficiency if $H(f(x)^*) \neq 0$ for the optimal x^*. Otherwise, if $H(f(x)^*) = 0$, we cannot ensure that x^* minimizes all individual objective functions. In this case we can only guarantee weak efficiency, see

Theorem 6. Consequently, we know that the individual objective functions are aimed at being minimized with the constraint $f(x) \leqq r$ and $H(f(x^*)) \neq 0$ for the optimal value x^*. Note that from $H(f(x)) > 0$ and $f(x) \leqq r$ it follows that $f(x) < r$. Thus, if $H(f(x))$ is positive, the constraint $f(x) \leqq r$ cannot be active in any component.

In general the constraint cannot become active when all feasible outcome vectors lie below the reference point, i.e., when r is chosen such that $f(x) < r$ for all $x \in \mathcal{X}$. In this case the constraint $f(x) \leqq r$ can be omitted. Furthermore, if $H(f(x^*)) \neq 0$ for the optimal x^*, it would be possible to exclude exactly one arbitrary but fixed index of the constraint (i.e., $f_l(x) \leq r_l$ for one $l \in \{1, \ldots, p\}$ can be omitted) since then it will be automatically satisfied at optimality due to the maximization of the product $H(f(x)) = \prod_{k=1}^{p} (r_k - f_k(x))$.

The following result shows that under the above assumptions the hypervolume scalarization (HV-SP) always generates a (weakly) efficient solution. This result is not new, however, we include a proof for the sake of completeness.

Theorem 6. *Let x^* be an optimal solution of the hypervolume scalarization problem* (HV-SP).

i) If $H(f(x^*)) \neq 0$, then x^* is efficient for the corresponding multiobjective optimization problem (MOP).

ii) If $H(f(x^*)) = 0$, then x^* is weakly efficient for the corresponding multiobjective optimization problem (MOP).

Proof. i) Suppose that the optimal solution x^* of (HV-SP) is not efficient for (MOP). Then there exists a solution $x' \in \mathcal{X}$ such that $f_k(x') \leq f_k(x^*)$ for all $k \in \{1, \ldots, p\}$ with at least one strict inequality. Since $H(f(x^*)) \neq 0$ and $f(x^*) \leqq r$ we have $r_k - f_k(x^*) > 0$ for all $k = 1, \ldots, p$. Thus, we have $r_k - f_k(x') \geq r_k - f_k(x^*) > 0$ for all $k \in \{1, \ldots, p\}$ with at least one strict inequality. It follows that

$$H(f(x')) = \prod_{k=1}^{p} (r_k - f_k(x')) > \prod_{k=1}^{p} (r_k - f_k(x^*)) = H(f(x^*)),$$

contradicting the assumption that x^* is optimal for (HV-SP).

ii) Suppose that the optimal solution x^* of (HV-SP) is not weakly efficient for (MOP). Then there exists a solution $x' \in \mathcal{X}$ such that $f_k(x') < f_k(x^*)$ for all $k \in \{1, \ldots, p\}$. Since $f(x^*) \leqq r$ we have $r_k - f_k(x^*) \geq 0$ for all $k = 1, \ldots, p$.

With $f_k(x') < f_k(x^*)$ for all $k \in \{1, \ldots, p\}$ it follows that $r_k - f_k(x') > 0$ for all $k \in \{1, \ldots, p\}$. All in all, we have

$$0 \neq H(f(x')) = \prod_{k=1}^{p} (r_k - f_k(x')) > \prod_{k=1}^{p} (r_k - f_k(x^*)) = H(f(x^*)) = 0,$$

contradicting the assumption that x^* is optimal for (HV-SP).

□

Conversely, we have the following:

Corollary 7. *If $x^* \in \mathcal{X}$ is an efficient solution of* (MOP)*, then it is optimal for* (HV-SP) *with reference point $r := f(x^*)$.*

Note that this is of rather theoretical interest since this choice of the reference point assumes that the corresponding efficient solution is already known. Practical choices for reference points in the context of biobjective shape optimization are discussed in Section 5.5.3.

Efficient numerical optimization procedures benefit from gradient information to obtain fast convergence. In this situation, we can take advantage of the fact that the gradient of the hypervolume indicator can be determined from the gradients of the individual objective functions using the product-rule of differentiation:

$$\nabla H(f(x)) = -\sum_{k=1}^{p} \prod_{\substack{l=1 \\ l \neq k}}^{p} (r_l - f_l(x)) \cdot \nabla f_k(x) \tag{3.3}$$

In Emmerich and Deutz, 2014 the properties and computational costs of the gradient of the hypervolume indicator $H(f(S))$ for a set of solutions $S \subset \mathcal{X}$ were analyzed. However, in this work we will only consider the hypervolume indicator $H(f(x))$ for one solution $x \in \mathcal{X}$. Thus, we observe that the gradient vector of the hypervolume indicator for one solution can be interpreted as a negatively weighted sum of the gradients of the individual objective functions. The weights depend on the component-wise distances from the current objective value vector to the reference point. We will analyze that in Section 3.4.1.

In the biobjective case ($p = 2$, i.e., $f = (f_1, f_2)^\top$) we have the hypervolume indicator

$$H(f(x)) = (r_1 - f_1(x)) \cdot (r_2 - f_2(x)) \tag{3.4}$$

and the gradient

$$\nabla H\left(f(x)\right) = -\begin{pmatrix} r_2 - f_2(x) \\ r_1 - f_1(x) \end{pmatrix}^{\top} \nabla f(x) \tag{3.5}$$

$$= -(r_2 - f_2(x)) \cdot \nabla f_1(x) - (r_1 - f_1(x)) \cdot \nabla f_2(x). \tag{3.6}$$

3.1 Related Literature

This literature review on the hypervolume indicator is taken from Schultes et al., 2021.

We refer exemplarily to Auger et al., 2009 and Yang et al., 2019 as two examples from this active research field. The hypervolume indicator also plays a prominent role as a quality indicator in the subset selection problem (see, for example, Guerreiro and Fonseca, 2020), i.e., when a subset of usually bounded cardinality is sought to represent the Pareto front. Theoretical properties as well as heuristic and exact solution approaches for subset selection problems were discussed, among many others, in Auger et al., 2009, 2012; Bringmann, Cabello, and Emmerich, 2017; Bringmann, Friedrich, and Klitzke, 2014; Guerreiro, Fonseca, and Paquete, 2016; Kuhn et al., 2016.

We particularly mention Emmerich and Deutz, 2014 for a detailed analysis of hypervolume gradients in the context of subset selection problems, a special case of which is considered in this work.

When restricting the search to individual solutions rather than solution sets, then the hypervolume indicator can be interpreted as a nonlinear scalarizing function, see, for example, Hernandez et al., 2020; Schulze et al., 2020; Touré et al., 2019. In the biobjective case, the hypervolume scalarization yields a quadratic objective function that has an intuitive geometrical interpretation. We note that quadratic scalarizations are also considered in Fliege, 2004 and in the context of reference point methods in Dandurand and Wiecek, 2016, and that in the biobjective case the hypervolume scalarization can be interpreted as a special case of these approaches. Furthermore, hypervolume scalarizations can be easily integrated into interactive approaches that are based on the iterative selection of reference points (like, for example, the Nimbus method, see Miettinen and Mäkelä, 1996). See Miettinen and Mäkelä, 2002 for a comparison of reference point methods illustrating how different methods elicitate the decision makers preferences.

The hypervolume scalarization shows some similarities with other reference point based methods like, e.g., compromise programming techniques which were introduced

by Bowman, 1976 and Freimer and Yu, 1976; Yu, 1973 in the context of group decisions. While compromise programming models typically aim at the minimization of the distance of an outcome vector from a given (utopian) reference point, e.g., based on ℓ_p-norms, the hypervolume scalarization aims at maximizing the volume dominated by an outcome vector w.r.t. a given reference point. Both approaches use nonlinear objective functions (except for compromise programming with ℓ_1-distances) and do not require additional constraints. See also Büsing et al., 2017 for recent research on reference point methods in the context of approximations of discrete multiobjective optimization problems.

3.2 Contour Lines of the Hypervolume Indicator

In the following we have a closer look at the contour lines of the hypervolume indicator. We want to motivate why the hypervolume method is not limited to the calculation of solutions in convex parts of the Pareto front. Corollary 7 already shown that in theory all efficient solutions can be computed by the hypervolume scalarization. However, $r := f(x^*)$, with x^* an efficient solution, is not a practicable choice for the reference point since we do not know the efficient set in advance.

This section is partly taken from Schultes et al., 2021 and extended from the biobjective case to the multiobjective case. We note an imprecise formulation in Schultes et al., 2021, p. 1212, stating that the (biobjective) hypervolume indicator H is a concave function, which is not true. The hypervolume indicator is neither convex nor concave. However, we will show here that the primary argument is that the contour lines of the hypervolume indicator are concave and thus, the hypervolume scalarization is not limited to calculate only supported efficient solutions.

The regions of the Pareto front which can be attained using a scalarization method are determined by the level sets and contour lines of the scalarizing function. Consider, for example, the contour lines of the weighted sum scalarization (hyperplanes) and of the weighted Chebyshev scalarization (hyperboxes with weight-dependent aspect ratio), respectively. While the former is restricted to supported non-dominated points, the latter has the potential to determine all non-dominated points, including the non-supported ones, see, for example, Ehrgott, 2005.

The hypervolume indicator H can be interpreted as a measure for the volume of an axis parallel box spanned by a point $y = f(x) \in \mathbb{R}^p$ and the reference point $r \in \mathbb{R}^p$. In the biobjective case ($p = 2$) we have an axis parallel rectangle and in the triobjective

case ($p = 3$) we have an axis parallel rectangular cuboid.

In the following, we rewrite the function H depending on $y \in \mathbb{R}^p$ instead of $f(x) \in \mathbb{R}^p$. The level set of H for the level $c \in \mathbb{R}_+$, with $\mathbb{R}_+$ is the set of positive real numbers, is given by

$$\mathcal{L}^H_=(c) \coloneqq \{y \in \mathbb{R}^p \,|\, H(y) = c, y < r\}\,. \tag{3.7}$$

We assume that $y < r$ since we are interested in points that are not dominated by the reference point r. Note that it follows from $y < r$ that $c > 0$. Thus, the level set is restricted to the domain $C = \{y \in \mathbb{R}^p \,|\, y < r\}$.

We define the index sets $\mathcal{I} \coloneqq \{1, \ldots, p\}$ and $\mathcal{I}_{-q} \coloneqq \mathcal{I} \backslash \{q\}$ for $q \in \{1, \ldots, p\}$ arbitrary but fixed. Solving the equation $H(y) = c$ for y_q with $q \in \{1, \ldots, p\}$ arbitrary but fixed, $y < r$ and $c > 0$ yields:

$$\begin{aligned}
& H(y) = c \\
\iff \quad & \prod_{k=1}^{p} (r_k - y_k) = c \\
\iff \quad & (r_q - y_q) \cdot \prod_{k \in \mathcal{I}_{-q}} (r_k - y_k) = c \\
\iff \quad & r_q - y_q = \frac{c}{\prod_{k \in \mathcal{I}_{-q}} (r_k - y_k)} \\
\iff \quad & -y_q = -r_q + \frac{c}{\prod_{k \in \mathcal{I}_{-q}} (r_k - y_k)} \\
\iff \quad & y_q = r_q - \frac{c}{\prod_{k \in \mathcal{I}_{-q}} (r_k - y_k)}
\end{aligned}$$

Note that from $y < r$ it follows that $(r_k - y_k) \neq 0$ for all $k = 1, \ldots, p$.

Let the domain C_{-q} given by

$$C_{-q} \coloneqq \left\{ y_{-q} \in \mathbb{R}^{p-1} \,\middle|\, y_k < r_k, k \in \mathcal{I}_{-q} \right\}. \tag{3.8}$$

Thus, the domain C_{-q} defines a subspace with the strict inequality $y_k < r_k$ for all $k \in \mathcal{I}_{-q}$ and the vector $y_{-q} \in \mathbb{R}^{p-1}$ is defined by the vector $y \in \mathbb{R}^p$ such that

$$y_{-q} = (y_1, \ldots, y_{q-1}, y_{q+1}, \ldots, y_p)^\top, \tag{3.9}$$

i.e., we consider vector y without index q. For a fixed level $c \in \mathbb{R}_+$ we can define a

function $s_q^c \colon C_{-q} \to \mathbb{R}$ with $q \in \{1, \ldots, p\}$ as

$$s_q^c\left(y_{-q}\right) \coloneqq r_q - \frac{c}{\prod_{k \in \mathcal{I}_{-q}} (r_k - y_k)} \tag{3.10}$$

Subsequently, the contour lines for $c \in \mathbb{R}_+$ can be described as

$$\mathcal{L}_{=}^{H}(c) = \left\{ \left(y_1, \ldots, y_{q-1}, s_q^c\left(y_{-q}\right), y_{q+1}, \ldots, y_p\right)^\top \in \mathbb{R}^p \,\middle|\, y_k < r_k \ \forall k \in \mathcal{I}_{-q} \right\}. \tag{3.11}$$

Lemma 8. *The function $s_q^c\left(y_{-q}\right)$ given by*

$$s_q^c\left(y_{-q}\right) \coloneqq r_q - \frac{c}{\prod_{k \in \mathcal{I}_{-q}} (r_k - y_k)} \tag{3.10}$$

with $q \in \{1, \ldots, p\}$ arbitrary but fixed and $c \in \mathbb{R}_+$ is concave on the domain $C_{-q} = \left\{ y_{-q} \in \mathbb{R}^{p-1} \,\middle|\, y_k < r_k, k \in \mathcal{I}_{-q} \right\}$.

Proof. Let $q \in \{1, \ldots, p\}$ be an arbitrary but fixed index. The partial derivatives and the second-order partial derivatives of s_q^c w.r.t. the components of $y_{-q} \in C_{-q}$ are given by

$$\frac{\partial s_q^c\left(y_{-q}\right)}{\partial y_m} = \frac{-c}{(r_m - y_m) \cdot \prod_{k \in \mathcal{I}_{-q}} (r_k - y_k)} \leq 0, \qquad m \in \mathcal{I}_{-q}; \tag{3.12}$$

$$\frac{\partial^2 s_q^c\left(y_{-q}\right)}{\partial y_m \partial y_n} = \frac{-c}{(r_m - y_m) \cdot (r_n - y_n) \cdot \prod_{k \in \mathcal{I}_{-q}} (r_k - y_k)} \leq 0, \quad m, n \in \mathcal{I}_{-q}, m \neq n; \tag{3.13}$$

$$\frac{\partial^2 s_q^c\left(y_{-q}\right)}{\partial y_m \partial y_m} = \frac{-2c}{((r_m - y_m))^2 \cdot \prod_{k \in \mathcal{I}_{-q}} (r_k - y_k)} \leq 0, \qquad m \in \mathcal{I}_{-q}. \tag{3.14}$$

The Hessian of $s_q^c\left(y_{-q}\right)$ can be written as

$$\mathbf{H}_{s_q^c}\left(y_{-q}\right) = \left(\theta_{m,n} \cdot z_{m,n}(y)\right)_{m,n \in \mathcal{I}_{-q}} \in \mathbb{R}^{(p-1)} \times \mathbb{R}^{(p-1)} \tag{3.15}$$

$$\text{with} \quad \theta_{m,n} \coloneqq \begin{cases} -2, & \text{if } m = n \\ -1, & \text{if } m \neq n \end{cases} \tag{3.16}$$

$$\text{and} \quad z_{m,n}(y) = \frac{c}{\prod_{k \in \mathcal{I}_{-q}} (r_k - y_k)} \cdot \frac{1}{(r_m - y_m) \cdot (r_n - y_n)}, \quad y \in C. \tag{3.17}$$

Note that $\theta_{m,n}$ and $z_{m,n}(y)$ are defined for all $m, n \in \mathcal{I}$ including $m, n = q$ since we need this for technical reasons to simplify the following computations. Furthermore, it holds that $z_{m,n}(y) \neq 0$ for all $y \in C$ since $c \in \mathbb{R}_+$ and $y < r$.

We will show that the Hessian $\mathbf{H}_{s_q^c}(y_{-q})$ is negative semi-definite on the domain C_{-q} by calculating the determinate of $\mathbf{H}_{s_q^c}(y_{-q})$ as well as the determinants of submatrices of $\mathbf{H}_{s_q^c}(y_{-q})$.

Towards this end, define the matrix $\mathbf{A}(y) \in \mathbb{R}^p \times \mathbb{R}^p$ as follows:

$$\mathbf{A}(y) = (\theta_{m,n} \cdot z_{m,n}(y))_{m,n \in \mathcal{I}} \tag{3.18}$$

$$= \begin{pmatrix} -2z_{1,1}(y) & -z_{1,2}(y) & \cdots & -z_{1,p-1}(y) & -z_{1,p}(y) \\ -z_{2,1}(y) & -2z_{2,2}(y) & \cdots & -z_{2,p-1}(y) & -z_{2,p}(y) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -z_{p-1,1}(y) & -z_{p-1,2}(y) & \cdots & -2z_{p-1,p-1}(y) & -z_{p-1,p}(y) \\ -z_{p,1}(y) & -z_{p,2}(y) & \cdots & -z_{p,p-1}(y) & -2z_{p,p}(y) \end{pmatrix} \tag{3.19}$$

If we delete row and column q from $\mathbf{A}(y)$, we get the Hessian matrix $\mathbf{H}_{s_q^c}(y_{-q})$, i.e., $\mathbf{H}_{s_q^c}(y_{-q})$ is a submatrix of $\mathbf{A}(y)$. We will first show how to calculate the determinant of $\mathbf{A}(y)$ and then explain how to calculate the determinants of submatrices of $\mathbf{A}(y)$.

We use Gaussian elimination to transform the matrix $\mathbf{A}(y)$ in a upper triangular matrix by performing a sequence of elementary row operations. Note that adding a scalar multiple of one row to another row does not change the determinant, see, for example, Liesen and Mehrmann, 2021, p. 103.

$$\begin{pmatrix} -2z_{1,1}(y) & -z_{1,2}(y) & -z_{1,3}(y) & \cdots & -z_{1,p}(y) \\ -z_{2,1}(y) & -2z_{2,2}(y) & -z_{2,3}(y) & \cdots & -z_{2,p}(y) \\ -z_{3,1}(y) & -z_{3,2}(y) & -2z_{3,3}(y) & \cdots & -z_{3,p}(y) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -z_{p,1}(y) & -z_{p,2}(y) & -z_{p,3}(y) & \cdots & -2z_{p,p}(y) \end{pmatrix} \begin{array}{l} \\ \leftarrow + \\ \leftarrow + \\ \\ \leftarrow + \\ \leftarrow + \end{array} \quad \begin{array}{l} \\ \frac{-1}{2} \cdot \frac{z_{2,1}(y)}{z_{1,1}(y)} \\ \frac{-1}{2} \cdot \frac{z_{3,1}(y)}{z_{1,1}(y)} \\ \\ \frac{-1}{2} \cdot \frac{z_{i,1}(y)}{z_{1,1}(y)} \\ \quad i=4,\ldots,p-1 \\ \frac{-1}{2} \cdot \frac{z_{p,1}(y)}{z_{1,1}(y)} \end{array} \tag{3.20}$$

$$\begin{pmatrix} -2z_{1,1}(y) & -z_{1,2}(y) & -z_{1,3}(y) & \cdots & -z_{1,p}(y) \\ 0 & \frac{-3}{2} z_{2,2}(y) & \frac{-1}{2} z_{2,3}(y) & \cdots & \frac{-1}{2} z_{2,p}(y) \\ 0 & \frac{-1}{2} z_{3,2}(y) & \frac{-3}{2} z_{3,3}(y) & \cdots & \frac{-1}{2} z_{3,p}(y) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \frac{-1}{2} z_{p,2}(y) & \frac{-1}{2} z_{p,3}(y) & \cdots & \frac{-3}{2} z_{p,p}(y) \end{pmatrix} \begin{array}{l} \\ \leftarrow + \\ \\ \\ \leftarrow + \end{array} \quad \begin{array}{l} \\ \frac{-1}{3} \cdot \frac{z_{3,2}(y)}{z_{2,2}(y)} \\ \\ \\ \frac{-1}{3} \cdot \frac{z_{i,2}(y)}{z_{2,2}(y)} \\ \quad i=4,\ldots,p-1 \\ \frac{-1}{3} \cdot \frac{z_{p,2}(y)}{z_{2,2}(y)} \end{array} \tag{3.21}$$

$$\begin{pmatrix} -2z_{1,1}(y) & -z_{1,2}(y) & -z_{1,3}(y) & \cdots & -z_{1,p}(y) \\ 0 & \frac{-3}{2}z_{2,2}(y) & \frac{-1}{2}z_{2,3}(y) & \cdots & \frac{-1}{2}z_{2,p}(y) \\ 0 & 0 & \frac{-4}{3}z_{3,3}(y) & \cdots & \frac{-1}{3}z_{3,p}(y) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \frac{-1}{3}z_{p,3}(y) & \cdots & \frac{-4}{3}z_{p,p}(y) \end{pmatrix} \quad \frac{-1}{4}\cdot\frac{z_{i,3}(y)}{z_{3,3}(y)},\ i=4,\dots,p-1 \ (+) \quad \frac{-1}{4}\cdot\frac{z_{p,3}(y)}{z_{3,3}(y)} \ (+) \tag{3.22}$$

The next rows will be transformed in the same way, i.e., row $i \in \{j+1,\dots,p\}$ will be transformed by adding $\frac{-z_{i,j}(y)}{(j+1)z_{j,j}(y)}$ times row j for $j = 4,\dots,p-1$, following to the last step:

$$\begin{pmatrix} -2z_{1,1}(y) & -z_{1,2}(y) & -z_{1,3}(y) & \cdots & -z_{1,p}(y) \\ 0 & \frac{-3}{2}z_{2,2}(y) & \frac{-1}{2}z_{2,3}(y) & \cdots & \frac{-1}{2}z_{2,p}(y) \\ 0 & 0 & \frac{-4}{3}z_{3,3}(y) & \cdots & \frac{-1}{3}z_{3,4}(y) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{-p}{(p-1)}z_{p,p}(y) \end{pmatrix} \quad \frac{-1}{p}\cdot\frac{z_{p,p-1}(y)}{z_{p-1,p-1}(y)} \ (+) \tag{3.23}$$

$$\begin{pmatrix} -2z_{1,1}(y) & -z_{1,2}(y) & -z_{1,3}(y) & \cdots & -z_{1,p}(y) \\ 0 & \frac{-3}{2}z_{2,2}(y) & \frac{-1}{2}z_{2,3}(y) & \cdots & \frac{-1}{2}z_{2,p}(y) \\ 0 & 0 & \frac{-4}{3}z_{3,3}(y) & \cdots & \frac{-1}{3}z_{3,4}(y) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{-(p+1)}{p}z_{p,p}(y) \end{pmatrix} \tag{3.24}$$

Since we only added scalar multiples of one row to another row, the determinants of the original matrix and of the transformed matrix are equal. Now we have an upper triangular matrix. Thus, the determinant can be calculated by its diagonal entries (Liesen and Mehrmann, 2021, p. 99). It follows that the determinant is given by

$$\det(\mathbf{A}(y)) = \prod_{k=1}^{p} (-1)\cdot\frac{k+1}{k}\cdot z_{k,k}(y) = (-1)^p\cdot(p+1)\cdot\prod_{k=1}^{p} z_{k,k}(y). \tag{3.25}$$

Now we define the submatrix $\mathbf{A}_{-q}(y)$ by deleting row q and column q from $\mathbf{A}(y)$. If we delete row q and column q from $\mathbf{A}(y)$, we get the Hessian matrix of s_q^c, i.e., $\mathbf{A}_{-q}(y) = \mathbf{H}_{s_q^c}(y_{-q})$. Additionally, we define the submatrix $\mathbf{A}_{-\mathcal{Q}}(y)$ for $\mathcal{Q} \subsetneq \mathcal{I}$ by deleting all columns and rows with indices $q' \in \mathcal{Q} \subsetneq \mathcal{I}$. The determinants of the submatrices $\mathbf{A}_{-\mathcal{Q}}(y)$ and $\mathbf{A}_{-q}(y)$, respectively, can be calculated in the same way as

above, see (3.20)–(3.24). It follows that the determinant of $\mathbf{A}_{-\mathcal{Q}}(y)$ is given by

$$\begin{aligned}\det\left(\mathbf{A}_{-\mathcal{Q}}(y)\right) &= \prod_{k=1}^{|\mathcal{I}_{-\mathcal{Q}}|}\left((-1)\cdot\frac{k+1}{k}\right)\cdot\prod_{k\in\mathcal{I}_{-\mathcal{Q}}} z_{k,k}(y)\\ &= (-1)^{|\mathcal{I}_{-\mathcal{Q}}|}\cdot(|\mathcal{I}_{-\mathcal{Q}}|+1)\cdot\prod_{k\in\mathcal{I}_{-\mathcal{Q}}} z_{k,k}(y)\end{aligned} \tag{3.26}$$

with $\mathcal{I}_{-\mathcal{Q}} = \mathcal{I}\backslash\mathcal{Q}$. Thus, the determinant of the Hessian matrix $\mathbf{H}_{s_q^c}(y_{-q})$ is given by

$$\det\left(\mathbf{H}_{s_q^c}(y_{-q})\right) = \det\left(\mathbf{A}_{-q}(y)\right) = (-1)^{p-1}\cdot p\cdot\prod_{k\in\mathcal{I}_{-q}} z_{k,k}(y). \tag{3.27}$$

Following Sylvester's minorant criterion, a real symmetric matrix is negative definite if and only if all even principal minors of this matrix are positive and all odd principal minors of this matrix are negative (see, for example, (Bhimasankaram and Rao, 2000, p. 342)).

The principal minors of $\mathbf{H}_{s_q^c}(y_{-q})$ are given by the determinants of the submatrices $\mathbf{A}_{-\mathcal{Q}}(y)$ with $\mathcal{Q}\subsetneq\mathcal{I}$ and $q\in\mathcal{Q}$. We have $z_{m,n}(y)>0$ for all $y\in C=\{y\in\mathbb{R}^p\,|\,y<r\}$ and $m,n\in\mathcal{I}$. Thus, all even principal minors of $\mathbf{H}_{s_q^c}(y_{-q})$ are positive and all odd principal minors of $\mathbf{H}_{s_q^c}(y_{-q})$ are negative, see (3.26). Consequently, the Hessian matrix $\mathbf{H}_{s_q^c}(y_{-q})$ is negative definite on the domain C_{-q} and thus, s_q^c is strictly concave for every $q\in\mathcal{I}$ with $c\in\mathbb{R}_+$. □

The reader may ask if there is a simpler way to show concavity of the level sets of H on the domain $C=\{y\in\mathbb{R}^p\,|\,y<r\}$ since their representation $y_q = s_q(y_{-q})$ considered on the domain C_q is a function composed of a linear function and the product of concave functions.

However we emphasize that the product of strictly convex (concave) functions is not strictly convex (concave) in general. This is even not the case if the functions have only positive (negative) values on the considered domain. Consider, for example, the one-dimensional functions $f(x)=x^{\frac{3}{2}}$ with $x>0$ and $h(x)=x^{-1}$ with $x>0$. These functions are strictly convex and have positive values on the considered domain $\{x\in\mathbb{R}\,|\,x>0\}$, but the product $f(x)\cdot h(x)=\sqrt{x}$ is strictly concave. This example is taken from Dietz, 2019.

We also note that there are special cases, e.g., in the one-dimensional case, where convexity (concavity) follows from appropriate conditions. Under the assumptions that two twice differentiable functions $f:\mathbb{R}\to\mathbb{R}$ and $h:\mathbb{R}\to\mathbb{R}$ are convex, have only

positive values and their first derivatives have the same sign, it can be shown that the product $f \cdot h$ is convex. For this consider the second derivative

$$(f \cdot h)'' = f'' \cdot h + 2f' \cdot h' + f \cdot h''. \tag{3.28}$$

We have assumed that $f(x) > 0$ and $h(x) > 0$ for all $x \in \mathbb{R}$. Since f and g are convex, it holds that $f''(x) \geq 0$ and $h''(x) \geq 0$ for all $x \in \mathbb{R}$, respectively. With this we have $f''(x) \cdot h(x) \geq 0$ and $f(x) \cdot h''(x) \geq 0$ for all $x \in \mathbb{R}$. We have also assumed that the first derivatives have the same sign, i.e., $f'(x) \cdot h'(x) \geq 0$ for all $x \in \mathbb{R}$. It follows that $2f'(x) \cdot h'(x) \geq 0$ for all $x \in \mathbb{R}$. Thus, the sum of these parts is also non-negative and therefore the product $f \cdot h$ is convex.

If we consider more than two objectives, i.e., $p \geq 3$, we have a multidimensional function for s_{-q} in Lemma 8. In this case the situation is more complicated. Consider the product $f \cdot h$ of two multidimensional functions $f : \mathbb{R}^n \to \mathbb{R}$ and $h : \mathbb{R}^n \to \mathbb{R}$ with $n \geq 2$ that are twice differentiable. Now we investigate whether there is a similar sufficient condition as the one discussed above for the one-dimensional case. Towards this end, assume that f and h are convex and that $f(x) > 0$ and $h(x) > 0$ for all $x \in \mathbb{R}^n$. We know that if the Hessian of a multidimensional function is positive semi-definite, then this function is convex. We consider Hessian $\mathbf{H}_{f \cdot h}$ of $f \cdot h$ and obtain

$$\mathbf{H}_{f \cdot h} = \mathbf{H}_f \cdot h + \nabla f \cdot (\nabla h)^\top + \nabla h \cdot (\nabla f)^\top + f \cdot \mathbf{H}_h. \tag{3.29}$$

For a matrix $A = A_1 + \cdots + A_\ell$ with $A_i \in \mathbb{R}^{n \times n}$ positive semi-definite for all $i = 1, \ldots, \ell$ we have by definition

$$A_i \text{ is positive semi-definite} \iff x^\top A_i x \geq 0 \quad \forall x \in \mathbb{R}^n,\ i = 1, \ldots, \ell. \tag{3.30}$$

Therefore, A also has to be positive semi-definite, because

$$x^\top A x = x^\top (A_1 + \cdots + A_\ell) x = x^\top A_1 x + \cdots + x^\top A_\ell x \geq 0 \quad \forall x \in \mathbb{R}^n. \tag{3.31}$$

Thus, we know that the sum of positive semi-definite matrices is also semi-positive definite (i.e., if $A_1, \ldots, A_\ell$ positive semi-definite, then $A = A_1 + \cdots + A_\ell$ is positive semi-definite). Note that the converse does not hold in general.

Since f and h are convex, the matrices $\mathbf{H}_f$ and $\mathbf{H}_h$ are positive semi-definite. We also assumed that $f(x) > 0$ and $h(x) > 0$ for all $x \in \mathbb{R}^n$. Thus, $\mathbf{H}_f \cdot h + f \cdot \mathbf{H}_h$ is positive semi-definite. It remains to show that the symmetric matrix $\nabla f \cdot (\nabla h)^\top +$

$\nabla h \cdot (\nabla f)^\top \in \mathbb{R}^{n \times n}$ is positive semi-definite. However, this matrix can be indefinite. Consider, for example, the case that $n = 2$ with the gradients of f and h given by $\nabla f(x) = (a(x), b(x))^\top$ and $\nabla h = (c(x), d(x))^\top$ with $x \in \mathbb{R}^n$ and $a, b, c, d : \mathbb{R}^n \to \mathbb{R}$. Then we have

$$\nabla f \cdot (\nabla h)^\top + \nabla h \cdot (\nabla f)^\top = \begin{pmatrix} ac & ad \\ bc & bd \end{pmatrix} + \begin{pmatrix} ca & cb \\ da & db \end{pmatrix} \tag{3.32}$$

$$= \begin{pmatrix} ac + ca & cb + ad \\ da + bc & db + bd \end{pmatrix} \tag{3.33}$$

$$= \begin{pmatrix} 2ac & cb + ad \\ cb + ad & 2bd \end{pmatrix}. \tag{3.34}$$

Thus, the determinant can be calculated as

$$\det\left(\nabla f \cdot (\nabla h)^\top + \nabla h \cdot (\nabla f)^\top\right) = 2ac \cdot 2bd - (cb + ad)(cb + ad) \tag{3.35}$$

$$= 4abcd - \left((cb)^2 + 2abcd + (ad)^2\right) \tag{3.36}$$

$$= 2abcd - (cb)^2 - (ad)^2 \tag{3.37}$$

$$= (-1) \cdot ((ad) - (bc))^2 \tag{3.38}$$

$$\leq 0. \tag{3.39}$$

If $(ad) - (bc) \neq 0$, i.e, if $(ad) \neq (bc)$ the matrix is indefinite. In summary, this does not lead to a sufficient condition for convexity. Therefore, the path chosen in the proof of Lemma 8 seems to be appropriate, although a large matrix has to be considered.

Corollary 7 already shows that every efficient solution x^* can be obtained by the hypervolume scalarization problem by choosing the reference point $r = f(x^*)$. With the concavity of the level sets in the objective space we have an additional argument that the hypervolume scalarization method is not limited to only calculating supported efficient solutions, even if the reference point is not chosen as $r = f(x^*)$. By interpreting the reference point as a scalarization parameter, in theory every efficient solution can be obtained as an optimal solution of a hypervolume scalarization with an appropriately chosen reference point. Note that Beume et al., 2009 showed a similar result for the biobjective case, namely that the level sets of the hypervolume indicator w.r.t. points y following the Pareto front $\mathcal{Y}_\mathrm{N}$ are concave. Nevertheless, we showed concavity of the level sets for points with $y < r$, not only for points $y \in \mathcal{Y}_\mathrm{N}$.

3.3 Interrelation Between Contour Lines and Reference Points

In the following section we use an observation taken from Schultes et al., 2021 and extend it to the multiobjective case. First, we consider the biobjective case.

The contour lines of the hypervolume indicator for a given reference point $r \in \mathbb{R}^2$ of a biobjective optimization problem are illustrated in the cone $r - \mathbb{R}^2_+$ in Figure 3.2a. It can be seen that their shape strongly depends on the level $c > 0$ of the corresponding level set. The lower the level c, the more the corresponding contour line resembles the contour line of a weighted l_∞ function, i.e., a weighted Chebyshev scalarization. The higher the level c, the more the contour line resembles that of a reference method with l_1 distances, that is a weighted sum scalarization with equal weights. In other words, the closer the reference point r is to a non-dominated outcome vector $\bar{y}$ in at least one component, the more the corresponding contour line resembles a weighted Chebyshev scalarization. On the other hand, the further the reference point is from a non-dominated outcome vector (in all components), the more the contour line approaches a weighted sum scalarization with equal weights.

More formally, consider the parameterization (3.10) of the contour line of the hypervolume indicator for the biobjective case w.r.t. y_1, i.e., $y_2 = s_2^c(y_1)$ with $y_1 < r_1$. For the level $\bar{c} = H(\bar{y})$ with $\bar{y} < r$ we have $\bar{c} > 0$ and $r_k \neq y_k$ for $k = 1, 2$. We assume for now that $y_1 < \bar{y}_1 < r_1$. Then we can see that

$$\begin{aligned}\lim_{r \to \bar{y}} s_2^{\bar{c}}(y_1) &= \lim_{r \to \bar{y}} \; r_2 - \frac{\bar{c}}{r_1 - y_1} \\ &= \lim_{r \to \bar{y}} \underbrace{r_2}_{\to \bar{y}_2} - \frac{\overbrace{(r_1 - \bar{y}_1)(r_2 - \bar{y}_2)}^{\to 0}}{\underbrace{r_1 - y_1}_{\to \bar{y}_1 - y_1 \neq 0}} = \bar{y}_2.\end{aligned}$$

This implies that for $r \to \bar{y}$ the function $s_2^{\bar{c}}(y_1)$ converges point-wise to a horizontal line passing through $\bar{y}$ with $y_1 < \bar{y}_1$. By using a parameterization $y_1 = s_1^c(y_2)$ it can be shown analogously that $r \to \bar{y}$ the function $s_1^{\bar{c}}(y_2)$ converges point-wise to a vertical line passing through $\bar{y}$ with $y_2 < \bar{y}_2 < r_2$.

This analysis can be generalized to the multiobjective case. Consider the parameterization $y_q = s_q^c(y_{-q})$ with $q \in \mathcal{I}$. For the level $\bar{c} = H(\bar{y})$ with $\bar{y} < r$ and with

$y_k < \bar{y}_k$ for all $k \in \mathcal{I}_{-q}$ we have

$$\lim_{r \to \bar{y}} s_q^{\bar{c}}(y_{-q}) = \lim_{r \to \bar{y}} \underbrace{r_q}_{\to \bar{y}_q} - \frac{\prod_{k=1}^{p} \overbrace{(r_k - \bar{y}_k)}^{\to 0}}{\prod_{k \in \mathcal{I}_{-q}} \underbrace{(r_k - y_k)}_{\to \underbrace{\bar{y}_k - y_k}_{\text{constant} \neq 0}}} = \bar{y}_q.$$

Thus, for $r \to \bar{y}$ the function $s_q^{\bar{c}}(y_{-q})$, with $y_k < \bar{y}_k$ for all $k \in \mathcal{I}_{-q}$, converges point-wise to a hyperplane defined by the normal $\mathbf{e}_q$ (the q-unit vector) and passing through $\bar{y}$.

Next, we consider the limit of moving the reference point infinitely far away from an arbitrary but fixed point $\bar{y} < r$. We first consider again the biobjective case: Let the reference point be given by $r = (\alpha_1, \alpha_2)^\top \hat{r}$ with $\hat{r} \in \mathbb{R}$. For $\bar{c} = H(\bar{y})$ and $y_1 < r_1$ we obtain that

$$\begin{aligned}
\lim_{\hat{r} \to \infty} s_2^{\bar{c}}(y_1) &= \lim_{\hat{r} \to \infty} \alpha_2 \hat{r} - \frac{(\alpha_1 \hat{r} - \bar{y}_1)(\alpha_2 \hat{r} - \bar{y}_2)}{\alpha_1 \hat{r} - y_1} \\
&= \lim_{\hat{r} \to \infty} \frac{\alpha_2 \hat{r}(\alpha_1 \hat{r} - y_1) - (\alpha_1 \hat{r} - \bar{y}_1)(\alpha_2 \hat{r} - \bar{y}_2)}{\alpha_1 \hat{r} - y_1} \\
&= \lim_{\hat{r} \to \infty} \frac{\alpha_1 \alpha_2 \hat{r}^2 - \alpha_2 \hat{r} y_1 - (\alpha_1 \alpha_2 \hat{r}^2 - \hat{r}(\alpha_2 \bar{y}_1 + \alpha_1 \bar{y}_2) + \bar{y}_1 \bar{y}_2)}{\alpha_1 \hat{r} - y_1} \\
&= \lim_{\hat{r} \to \infty} \frac{\hat{r}(\alpha_2 \bar{y}_1 + \alpha_1 \bar{y}_2 - \alpha_2 y_1) - \bar{y}_1 \bar{y}_2}{\alpha_1 \hat{r} - y_1} \\
&= \lim_{\hat{r} \to \infty} \frac{\frac{\hat{r}}{\hat{r}}(\alpha_2 \bar{y}_1 + \alpha_1 \bar{y}_2 - \alpha_2 y_1) - \frac{1}{\hat{r}} \bar{y}_1 \bar{y}_2}{\alpha_1 \frac{\hat{r}}{\hat{r}} - \frac{1}{\hat{r}} y_1} \\
&= \frac{\alpha_2 \bar{y}_1 + \alpha_1 \bar{y}_2 - \alpha_2 y_1}{\alpha_1} \\
&= -\frac{\alpha_2}{\alpha_1}\Big(y_1 - \bar{y}_1\Big) + \bar{y}_2.
\end{aligned}$$

The function $s_2^{\bar{c}}(y_1)$ converges point-wise to a line with slope $-\frac{\alpha_2}{\alpha_1}$ and passing through $\bar{y}$.

Generalized to the multiobjective case we obtain the following result:

Lemma 9. *Let $\bar{y} \in \mathbb{R}^p$ with $\bar{y} < r$ and let $s_q^{\bar{c}}(y_{-q})$ be the contour line representation for level $\bar{c} = H(\bar{y}) \in \mathbb{R}_+$ with $q \in \mathcal{I}$ arbitrary but fixed. For $r = (\alpha_1, \ldots, \alpha_p)^\top \hat{r} \in \mathbb{R}^p$, and $\hat{r} \to \infty$ we obtain*

$$\lim_{r \to \infty} s_q^{\bar{c}}(y_{-q}) = \lim_{\hat{r} \to \infty} \alpha_q \hat{r} - \frac{\prod_{k \in \mathcal{I}} (\alpha_k \hat{r} - \bar{y}_k)}{\prod_{k \in \mathcal{I}_{-q}} (\alpha_k \hat{r} - y_k)} = \sum_{k \in \mathcal{I}_{-q}} -\frac{\alpha_q}{\alpha_k}(y_k - \bar{y}_k) + \bar{y}_q.$$

Proof. We have

$$s_q^c\,(y_{-q}) = \alpha_q\hat{r} - \frac{\prod_{k\in\mathcal{I}}(\alpha_k\hat{r}-\bar{y}_k)}{\prod_{k\in\mathcal{I}_{-q}}(\alpha_k\hat{r}-y_k)} = \frac{\alpha_q\hat{r}\prod_{k\in\mathcal{I}_{-q}}(\alpha_k\hat{r}-y_k)-\prod_{k\in\mathcal{I}}(\alpha_k\hat{r}-\bar{y}_k)}{\prod_{k\in\mathcal{I}_{-q}}(\alpha_k\hat{r}-y_k)}.$$

These products can be expanded, see (3.40), (3.41) and (3.42) below. We denote the power set of the index sets $\mathcal{I}=\{1,\dots,p\}$ and $\mathcal{I}_{-q}=\mathcal{I}\backslash\{q\}$ by $P(\mathcal{I})$ and $P(\mathcal{I}_{-q})$, respectively. The cardinality of an index set $\mathcal{J}$ is denoted by $|\mathcal{J}|$.

$$\prod_{k\in\mathcal{I}_{-q}}(\alpha_k\hat{r}-y_k) = \left(\prod_{k\in\mathcal{I}_{-q}}\alpha_k\right)\cdot\hat{r}^{p-1} + \left(\prod_{k\in\mathcal{I}_{-q}}\alpha_k\right)\cdot\sum_{\ell=1}^{p-1}\Bigg((-1)^\ell\cdot\hat{r}^{p-1-\ell}\cdot\underbrace{\sum_{\substack{\mathcal{J}\in P(\mathcal{I}_{-q})\\|\mathcal{J}|=\ell}}\prod_{j\in\mathcal{J}}\frac{y_j}{\alpha_j}}_{=(\mathcal{S}_1)_\ell}\Bigg) \tag{3.40}$$

$$\alpha_q\hat{r}\cdot\prod_{k\in\mathcal{I}_{-q}}(\alpha_k\hat{r}-y_k) = \left(\prod_{k\in\mathcal{I}}\alpha_k\right)\cdot\hat{r}^{p} + \left(\prod_{k\in\mathcal{I}}\alpha_k\right)\cdot\sum_{\ell=1}^{p-1}\Bigg((-1)^\ell\cdot\hat{r}^{p-\ell}\cdot\underbrace{\sum_{\substack{\mathcal{J}\in P(\mathcal{I}_{-q})\\|\mathcal{J}|=\ell}}\prod_{j\in\mathcal{J}}\frac{y_j}{\alpha_j}}_{=(\mathcal{S}_1)_\ell}\Bigg) \tag{3.41}$$

$$\prod_{k\in\mathcal{I}}(\alpha_k\hat{r}-\bar{y}_k) = \left(\prod_{k\in\mathcal{I}}\alpha_k\right)\cdot\hat{r}^{p} + \left(\prod_{k\in\mathcal{I}}\alpha_k\right)\cdot\sum_{\ell=1}^{p}\Bigg((-1)^\ell\cdot\hat{r}^{p-\ell}\cdot\underbrace{\sum_{\substack{\mathcal{J}\in P(\mathcal{I})\\|\mathcal{J}|=\ell}}\prod_{j\in\mathcal{J}}\frac{\bar{y}_j}{\alpha_j}}_{=(\mathcal{S}_2)_\ell}\Bigg) \tag{3.42}$$

The factors $(\mathcal{S}_1)_\ell$ and $(\mathcal{S}_2)_\ell$ are independent of $\hat{r}$. All in all, we have:

$$\lim_{\hat{r}\to\infty} s_q^c(y_{-q}) = \lim_{\hat{r}\to\infty}\frac{\alpha_q\hat{r}\prod_{k\in\mathcal{I}_{-q}}(\alpha_k\hat{r}-y_k)-\prod_{k\in\mathcal{I}}(\alpha_k\hat{r}-\bar{y}_k)}{\prod_{k\in\mathcal{I}_{-q}}(\alpha_k\hat{r}-y_k)} \tag{3.43}$$

$$= \lim_{\hat{r}\to\infty}\frac{\left(\prod_{k\in\mathcal{I}}\alpha_k\right)\cdot\left(\sum_{\ell=1}^{p-1}\left((-1)^\ell\hat{r}^{p-\ell}\cdot(\mathcal{S}_1)_\ell\right)-\sum_{\ell=1}^{p}\left((-1)^\ell\hat{r}^{p-\ell}\cdot(\mathcal{S}_2)_\ell\right)\right)}{\left(\prod_{k\in\mathcal{I}_{-q}}\alpha_k\right)\cdot\hat{r}^{p-1}+\left(\prod_{k\in\mathcal{I}_{-q}}\alpha_k\right)\cdot\sum_{\ell=1}^{p-1}(-1)^\ell\hat{r}^{p-1-\ell}\cdot(\mathcal{S}_1)_\ell} \tag{3.44}$$

$$= \alpha_q\lim_{\hat{r}\to\infty}\frac{\left(\sum_{\ell=1}^{p-1}\left((-1)^\ell\hat{r}^{p-\ell}\cdot(\mathcal{S}_1)_\ell\right)-\sum_{\ell=1}^{p}\left((-1)^\ell\hat{r}^{p-\ell}\cdot(\mathcal{S}_2)_\ell\right)\right)}{\hat{r}^{p-1}+\sum_{\ell=1}^{p-1}(-1)^\ell\hat{r}^{p-1-\ell}\cdot(\mathcal{S}_1)_\ell} \tag{3.45}$$

$$= \alpha_q\lim_{\hat{r}\to\infty}\frac{\sum_{\ell=1}^{p-1}(-1)^\ell\overbrace{\frac{\hat{r}^{p-\ell}}{\hat{r}^{p-1}}}^{\substack{=1 \text{ if } \ell=1\\ \to 0 \text{ if } \ell>1}}\cdot(\mathcal{S}_1)_\ell-\sum_{\ell=1}^{p}(-1)^\ell\overbrace{\frac{\hat{r}^{p-\ell}}{\hat{r}^{p-1}}}^{\substack{=1 \text{ if } \ell=1\\ \to 0 \text{ if } \ell>1}}\cdot(\mathcal{S}_2)_\ell}{\underbrace{\frac{\hat{r}^{p-1}}{\hat{r}^{p-1}}}_{=1}+\sum_{\ell=1}^{p-1}(-1)^\ell\underbrace{\frac{\hat{r}^{p-1-\ell}}{\hat{r}^{p-1}}}_{\to 0}\cdot(\mathcal{S}_1)_\ell} \tag{3.46}$$

$$= \alpha_q \left((\mathcal{S}_2)_{\ell=1} - (\mathcal{S}_1)_{\ell=1} \right) \tag{3.47}$$

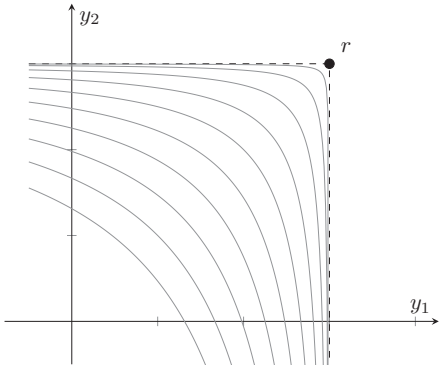
$$= \alpha_q \left(\sum_{k \in \mathcal{I}} \frac{\bar{y}_k}{\alpha_k} - \sum_{k \in \mathcal{I}_{-q}} \frac{y_k}{\alpha_k} \right) \tag{3.48}$$

□

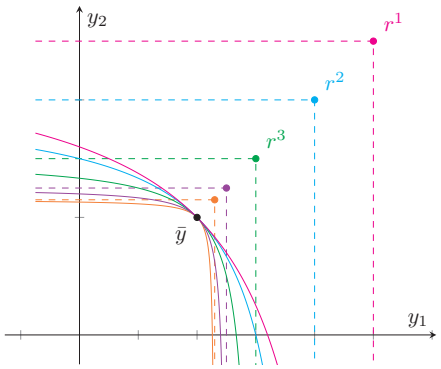
As a result, the level set $\mathcal{L}^H_{=}(\bar{c})$ converges for $r = (\alpha_1, \ldots, \alpha_p)^\top \hat{r}$, $\hat{r} \to \infty$ point-wise to a hyperplane with normal vector $(\frac{1}{\alpha_1}, \ldots, \frac{1}{\alpha_p})^\top \in \mathbb{R}^p$ and position vector $\bar{y} \in \mathbb{R}^p$:

$$\begin{aligned} \mathcal{L}^H_{=}(c) &\overset{\hat{r} \to \infty}{\longrightarrow} \left\{ y \in \mathbb{R}^p \;\middle|\; \frac{y_1}{\alpha_1} + \cdots + \frac{y_p}{\alpha_p} = \sum_{k=1}^{p} \frac{\bar{y}_k}{\alpha_k} \right\} \\ &= \left\{ y \in \mathbb{R}^p \;\middle|\; \left\langle \left(\frac{1}{\alpha_1}, \ldots, \frac{1}{\alpha_p} \right)^\top, y - \bar{y} \right\rangle = 0 \right\} \end{aligned}$$

The contour lines in the biobjective case passing through a given point $\bar{y} \in \mathbb{R}^2$, that are generated by different reference points $r > \bar{y}$, are illustrated in Figure 3.2b.



(a) Contour lines for a fixed reference point $r \in \mathbb{R}^2$ in $r - \mathbb{R}^2_{>}$.



(b) Contour lines passing through $\bar{y}$ w.r.t. different reference points $r > \bar{y}$.

Figure 3.2: Contour lines of the hypervolume indicator for the biobjective case, see Schultes et al., 2021.

3.4 Comparison

3.4.1 Hypervolume Scalarization vs Weighted Sum Scalarization

As in Schultes et al., 2021 we compare the hypervolume scalarization to the weighted sum scalarization. To interrelate hypervolume indicator for one solution with weighted sum scalarizations, we will analyze the level sets as well as the gradients since we will later use the steepest descent method for these scalarizations.

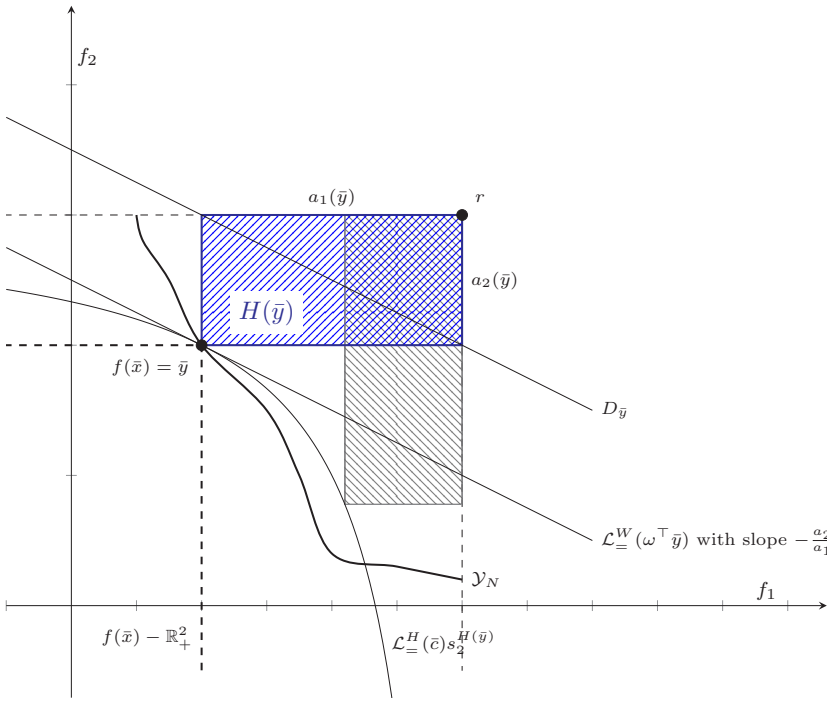


Figure 3.3: A non-dominated point which is suboptimal w.r.t. the hypervolume indicator.

In the following, let $r = (r_1, \ldots, r_p)^\top$ be a fixed reference point and let $\bar{x} \in \mathcal{X}$ be a strictly feasible solution of the hypervolume scalarization problem (HV-SP), i.e., $\bar{y} = f(\bar{x}) < r$.

To simplify the notation in the multiobjective case, we denote the side lengths of the hypervolume orthotope spanned by a point $y = (y_1, \ldots, y_p)^\top$ and the reference point r with $a_k(y) \coloneqq r_k - y_k$ for all $k = 1, \ldots, p$. Now the hypervolume induced by y

can be rewritten as

$$H(y) = \prod_{k=1}^{p} a_k(y). \tag{3.49}$$

Remember that this corresponds to the volume of the orthotope spanned by the point y and the reference point r. Thus the side length of this orthotope are given by $a(y) = (a_1(y), \ldots, a_p(y))^\top \in \mathbb{R}$.

We first have a look at the level sets. On the one hand we know that the level sets of the weighted sum scalarization are linear functions. The level sets of the hypervolume indicator on the other hand are concave functions. We now have a look at the tangent space of the hypervolume indicator level sets for at a specific level $\bar{c} = H(\bar{y})$ at point $\bar{y}$.

In Section 3.2 we have shown that the level set for the hypervolume indicator for a level $c \in \mathbb{R}$ can be described by

$$\mathcal{L}_{=}^{H}(c) = \left\{ \left(y_1, \ldots, y_{q-1}, s_q^c\left(y_{-q}\right), y_{q+1}, \ldots, y_p\right)^\top \in \mathbb{R}^p \right\}. \tag{3.11}$$

with the parameterization $s^c : H(y) = c$

$$s_q^c\left(y_{-q}\right) = r_q - \frac{c}{\prod_{k \in \mathcal{I}_{-q}} (r_k - y_k)} \tag{3.10}$$

$$s_q^c\left(y_{-q}\right) = r_q - \frac{c}{\prod_{k \in \mathcal{I}_{-q}} a_k(y)}. \tag{3.50}$$

The gradient of $s_q^c\left(y_{-q}\right)$ was calculated in (3.12). Thus the gradient of the contour line is given by

$$\nabla s_q^c\left(y_{-q}\right) = \left(\frac{-c}{a_k(y) \cdot \prod_{m \in \mathcal{I}_{-q}} a_m(y)} \right)_{k \in \mathcal{I}_{-q}} \tag{3.51}$$

For level $\bar{c} = H(\bar{y}) = \prod_{k=1}^{p} a_k(\bar{y})$, we get

$$\nabla s_q^{\bar{c}}\left((\bar{y}_k)_{k \in \mathcal{I}_{-q}}\right) = \left(\frac{-\prod_{m=1}^{p} a_m(\bar{y})}{a_k(\bar{y}) \cdot \prod_{m \in \mathcal{I}_{-q}} a_m(\bar{y})} \right)_{k \in \mathcal{I}_{-q}} \tag{3.52}$$

$$= \left(-\frac{a_q(\bar{y})}{a_k(\bar{y})} \right)_{k \in \mathcal{I}_{-q}} \tag{3.53}$$

Thus we have the tangent space of $\mathcal{L}_{=}^{H}(\bar{c})$ in point $\bar{y}$ given by

$$T_q^H(y_{-q}) = \bar{y}_q + \sum_{k \in \mathcal{I}_{-q}} -\frac{a_q(\bar{y})}{a_k(\bar{y})}(y_k - \bar{y}_k) \tag{3.54}$$

$$= \bar{y}_q - a_q(\bar{y}) \sum_{k \in \mathcal{I}_{-q}} \frac{(y_k - \bar{y}_k)}{a_k(\bar{y})} \tag{3.55}$$

The tangent space can also be written in point-normal form, with the normal vector given by the gradient of $H(y)$, i.e. $n_T = \sum_{k \in \mathcal{I}} \prod_{l \in \mathcal{I}_{-k}} a_l(\bar{y})$.

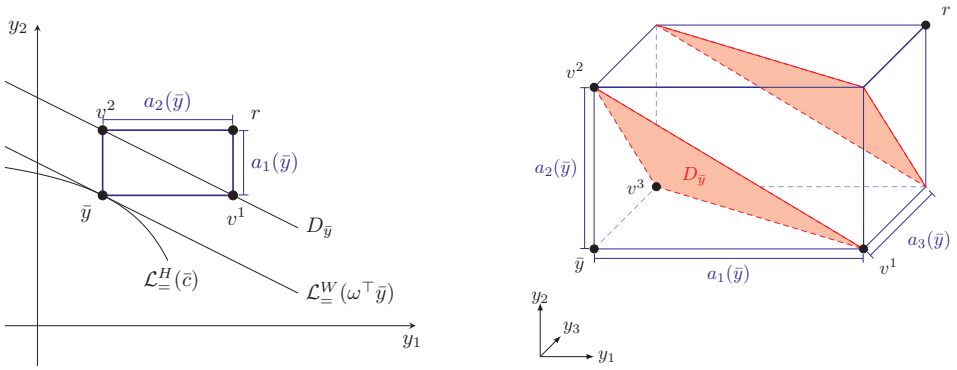
$$T^H : 0 = \sum_{k \in \mathcal{I}} \prod_{l \in \mathcal{I}_{-k}} a_l(\bar{y})(y_k - \bar{y}_k) \tag{3.56}$$

It can also be shown, that the diagonal $D_{\bar{y}}$, given by the points $v^i = \bar{y} + a_i(\bar{y})e_i$ with e_i the standard unit vector, i.e., $v^i = \left(v_1^i, \ldots, v_p^i\right)^\top \in \mathbb{R}^p$ with $v_k^i = \bar{y}_k$ for all $i \neq k$ and $v_k^i = r_k$, is parallel to the tangent space (3.56). The diagonal space is the hyperplane spanned by the vectors d^i, $i = 1, \ldots, p-1$ with $d^i = v^p - v^i$. In Figure 3.4 are the diagonals for a objective space with two (Figure 3.4a) and three (Figure 3.4b) objectives are visualized.

The normal vector n_D can be calculated by a generalization of the cross product:

$$n_D = d^1 \times \cdots \times d^{p-1} = \left(\begin{pmatrix} \bar{y}_1 \\ \vdots \\ \bar{y}_{p-1} \\ r_p \end{pmatrix} - \begin{pmatrix} r_1 \\ \bar{y}_2 \\ \vdots \\ \bar{y}_p \end{pmatrix} \right) \times \cdots \times \left(\begin{pmatrix} \bar{y}_1 \\ \vdots \\ \bar{y}_{p-1} \\ r_p \end{pmatrix} - \begin{pmatrix} \bar{y}_1 \\ \vdots \\ \bar{y}_{p-2} \\ r_{p-1} \\ \bar{y}_p \end{pmatrix} \right) \tag{3.57}$$

$$= \begin{pmatrix} -a_1(\bar{y}) \\ 0 \\ \vdots \\ 0 \\ a_p(\bar{y}) \end{pmatrix} \times \cdots \times \begin{pmatrix} 0 \\ \vdots \\ 0 \\ -a_{p-1}(\bar{y}) \\ a_p(\bar{y}) \end{pmatrix} \tag{3.58}$$



(a) $p = 2$ (b) $p = 3$

Figure 3.4: Visualization of the diagonals for in the biobjective and triobjective case.

$$= \det \begin{pmatrix} e_1 & -a_1(\bar{y}) & 0 & \dots & 0 \\ \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ e_{p-1} & 0 & \dots & 0 & -a_{p-1}(\bar{y}) \\ e_p & a_p(\bar{y}) & \dots & \dots & a_p(\bar{y}) \end{pmatrix} \tag{3.59}$$

$$= \sum_{q \in \mathcal{I}} (-1)^{p-1} \cdot e_q \cdot \prod_{l \in \mathcal{I}_{-q}} a_l(\bar{y}) \tag{3.60}$$

$$= (-1)^{p-1} \cdot \sum_{q \in \mathcal{I}} e_q \cdot \prod_{l \in \mathcal{I}_{-q}} a_l(\bar{y}) \tag{3.61}$$

$$= (-1)^{p-1} \cdot \left(\prod_{l \in \mathcal{I}_{-1}} a_l(\bar{y}), \dots, \prod_{l \in \mathcal{I}_{-p}} a_l(\bar{y}) \right)^\top \tag{3.62}$$

As a result, we have $n_D = \alpha \cdot n_T$ with a scalar $\alpha \in \mathbb{R}$, i.e. the diagonal space spanned by d^i with $i \in \mathcal{I}_{-p}$ and the the tangent space (3.56) are parallel. All in all, the slope of the tangent space is defined by the ratio of the side lengths $a_k(\bar{y})$ with $k = 1, \dots, p$, and is the same as the slope of the diagonal space $D_{\bar{y}}$.

The level sets of the weighted sum scalarization are given by

$$\mathcal{L}^W_=(c) = \left\{ (y_1, \dots, y_p)^\top \in \mathbb{R}^p \,\middle|\, \omega^\top y = c \right\}. \tag{3.63}$$

Therefore the normal vector of these level sets are given by $n_W = \omega \in \mathbb{R}^p$. If we

choose $w_k = \frac{1}{a_k(\bar{y})}$,or $w_k = \prod_{l \in \mathcal{I}_{-k}} a_l(\bar{y})$, see (3.56), for the weights, we can see, that the level set of the weighted sum $\mathcal{L}^W_=(c)$ with $c \in \mathbb{R}$ is parallel to the diagonal $D_{\bar{y}}$ and the tangent space T^H of $\mathcal{L}^H_=(\bar{c})$ in point $\bar{y}$, because there have the same normal direction n_D and n_T. For level $c = \sum_{k \in \mathcal{I}} \frac{1}{a_k(\bar{y})} \cdot \bar{y_k}$ the weighted sum scalarization is equal to this tangent space. Note that the weights of the weighted sum scalarization are often normalized to $\sum_{k=1}^{p} \omega_k = 1$ for simplification, but this can be omitted without affecting the properties of the weighted sum scalarization method.

All in all, that tangent space of hypervolume level sets at a point $\bar{y}$ are equal or parallel to the level sets of the weighted sum scalarization with appropriate chosen weights that depend on the distance of $\bar{y}$ to the reference point r.

Furthermore, choosing the weights $w_k = \prod_{l \in \mathcal{I}_{-k}} a_l(\bar{y})$ the weighted sum scalarization and the hypervolume scalarization shares the same direction of largest improvement in point $\bar{y} = f(\bar{x})$, i.e., $-\nabla W(\bar{x}) = \nabla H(f(\bar{x})$, because

$$-\nabla W(\bar{x}) = -\sum_{k=1}^{p} \prod_{l \in \mathcal{I}_{-k}} a_l(\bar{y}) \nabla f_k(\bar{x}) = \nabla H\left(f(\bar{x})\right) \tag{3.64}$$

Thus, the negative gradient of the hypervolume indicator for a fixed solution $\bar{x}$ is a weighted sum of the gradients of the single-objective gradients, with weights depending on the current side length defined by reference point r and $\bar{y} = f(\bar{x})$.

In the same way as in Schultes et al., 2021 we can derive, that if an iterative ascent algorithm for the hypervolume scalarization (HV-SP) is initialized with a strictly feasible starting solution $x^{(0)} \in \mathcal{X}$, i.e., $f(x^{(0)}) < r$, then moving in the direction of steepest ascent with an appropriate step length control ensures that the next iterate $x^{(1)}$ is also strictly feasible, i.e., $f_k(x^{(1)}) < r_k$ for $k = 1, \ldots, p$. Note that only local descent is guaranteed. Hence, the constraints $f_k(x) \leq r_k$, $k = 1, \ldots, p$, can be omitted in (HV-SP) when starting with a strictly feasible starting solution. Note that for any strictly feasible solution $x \in \mathcal{X}$ the side lengths of the hypervolume rectangle $a_k(f(x))$ correspond to the slackness of the constraints $f_k(\bar{\Omega}) \leq r_k$ for all $k = 1, \ldots, p$. This is reflected in the gradient direction $\nabla H(f(\bar{x}))$ at $f(\bar{x})$ that is composed of a weighted sum of the gradients of the individual objective functions $\nabla f_k(\bar{x})$, $k = 1, \ldots, p$. The ratio of the side lengths of the hypervolume rectangle defines the relative contribution of $\nabla f_k(\bar{x})$ to the ascent direction $\nabla H(f(\bar{x}))$. If this ratio is very large (very small, respectively), which is the case when one side of the hypervolume rectangle is considerably larger than the other, then the opposite objective function is emphasized in the computation of $\nabla H(f(\bar{x}))$.

3.4.2 Hypervolume Scalarization vs Multiobjective Descent Algorithms

A comparison of the hypervolume scalarization method to the biobjective descent algorithm was also done in Schultes et al., 2021, pp. 1215-1216, these results can be one to one adapted to the multiobjective case.

Multiobjective gradient descent algorithms were proposed in Désidéri, 2012; Fliege, Vaz, and Vicente, 2019; Fliege and Svaiter, 2000. In the minimization case, they can be interpreted as iterative optimization processes with search direction

$$d^{(i)} \coloneqq -{\omega^{(i)}}^{\top} \nabla f\left(x^{(i)}\right) = -\sum_{k=1}^{p} \omega_k^{(i)} \nabla f_k\left(x^{(i)}\right)$$

in iteration i with weights $\omega_k^{(i)} \geq 0$ for all $k = 1, \ldots, p$. The search direction $d^{(i)}$ is thus induced by a positive linear combination of the gradients of the individual objective functions with weights $\omega_k^{(i)}$.

These weights are chosen adaptively to achieve the *steepest* descent w.r.t. all objectives in each iteration. This is in contrast to applying a single-objective gradient descent algorithm to the weighted sum scalarization (WS-SP), in which the weights are predefined and thus fixed throughout all iterations of the optimization procedure. We argue that performing gradient ascent steps w.r.t. the hypervolume indicator is somewhat similar to applying a multiobjective gradient descent algorithm: The weights $\omega_k^{(i)} = \prod_{\substack{l=1 \\ l \neq k}} a_l\left(f\left(x^{(i)}\right)\right) = \prod_{\substack{l=1 \\ l \neq k}} \left(r_l - f_l\left(x^{(i)}\right)\right)$ of the gradients of the individual objective functions are chosen adaptively, depending on the side lengths of the hypervolume orthotope induced by the current iterate, c.f. (3.3).

While each iterate $x^{(i)}$ in a multiobjective gradient descent algorithm dominates all previous iterates $x^{(i-\ell)}$ for all $\ell = 1, \ldots, k$, i.e., $f\left(x^{(i)}\right) \leqslant f\left(x^{(i-\ell)}\right)$, this is in general not the case when applying a single-objective ascent algorithm to the hypervolume scalarization or a single-objective descent algorithm to the weighted sum scalarization, respectively. Indeed, only single-objective improvements can be guaranteed with respect to the hypervolume indicator and the weighted sum objective, i.e., $H\left(x^{(i)}\right) \geq H\left(x^{(i-\ell)}\right)$ and $\omega^{\top} f\left(x^{(i)}\right) \leq \omega^{\top} f\left(x^{(i-\ell)}\right)$, respectively. This can be seen in Figure 3.3 when comparing the dominance cone $f(\bar{x}) - \mathbb{R}^2_{\geqslant}$ (which contains the image of the next iterate in a multiobjective descent algorithm), the set of improving points for the hypervolume indicator given by $\mathcal{L}^H_{\leq}(\bar{c}) = \mathcal{L}^H_{=}(\bar{c}) - \mathbb{R}^2_{\geqq} = \{y \in \mathbb{R}^2 \colon (r_1 - y_1) \cdot (r_2 - y_2) \geq H(f(\bar{x}))\}$, and the set of improving points for the weighted

sum objective given by $\mathcal{L}^W_{\leq}(\omega^\top f(\bar{x})) = \mathcal{L}^W_{=}(\omega^\top f(\bar{x})) - \mathbb{R}^2_{\geqq} = \{y \in \mathbb{R}^2 \colon \omega^\top y \leq \omega^\top f(\bar{x})\}$ (with $\omega^\top = (a_2(\bar{y}), a_1(\bar{y}))$). Since the dominance cone $f(\bar{x}) - \mathbb{R}^2_{\geqslant}$ is a proper subset of the respective sets of improving points, multiobjective gradient descent algorithms are more restrictive regarding the choice of the next iterate. The weighted sum scalarization is in this regard the most flexible approach, allowing for the deterioration of individual objective functions as long as the weighted sum value improves. In this respect, we can say that the hypervolume indicator function compromises between the other two approaches.

Chapter 4

PDE Constrained Shape Optimization

The aim of this chapter is to give a brief introduction to PDE constrained shape optimization. This chapter is based on Haslinger and Mäkinen, 2003. For an extensive introduction to shape optimization, we refer, among others, to Allaire, 2002; Bucur and Buttazzo, 2005; Haslinger and Mäkinen, 2003; Sokolowski and Zolesio, 1992.

Shape optimization is a topic in the field of *structural optimization*, an important branch in computational mechanics. Structural optimization can be seen as a generic term for the optimization of basic structural properties such as the shape, size, topology or material properties of a considered mechanical component. It deals with the task of setting up a mathematical model to describe the behavior of the component. The goal is to find a structure that meets the desired properties.

In structural optimization the task of finding an optimal shape, in addition to adjusting the material properties, can be divided into three different categories: *sizing optimization*, *shape optimization* and *topology optimization*. In sizing optimization the shape of the component is predefined and only its size is subject to the optimization. In shape optimization, which we will focus on, the shape (and size) of the component is subject to optimization without changing the topology of the shape. In topology optimization, however, the topology of the shape can also be changed and is part of the optimization process, e.g., holes could be added to the shape.

In this thesis we will consider shape optimization problems. We want to find an optimal shape Ω that minimizes a given objective function with respect to certain

constraints. Thereby, we will consider objectives that depend on the solution of a partial differential equation (PDE). If not otherwise stated, we call this PDE the state equation $\mathcal{P}$ and its solution is denoted by $u(\Omega)$. We assume that the PDE and its solution are uniquely defined by Ω. Thus, a PDE constrained objective function J not only depends on the shape Ω, but also on the solution $u(\Omega)$ of the given state equation. In general, a shape optimization problem is given by

$$\begin{aligned} \min \quad & J(\Omega, u(\Omega)) \\ \text{s. t.} \quad & \Omega \in \mathcal{O}^{\text{ad}}, \\ & u(\Omega) \text{ solves the state equation } \mathcal{P}, \end{aligned} \tag{4.1}$$

where $\mathcal{O}^{\text{ad}}$ is the set of feasible shapes. The existence of optimal shapes is considered in Allaire, 2002; Chirkov et al., 2018; Fujii, 1988; Haslinger and Mäkinen, 2003 among others.

For the numerical solution of the state equation $\mathcal{P}$ and evaluation the objective function we will use finite element analysis. Therefore, the shape Ω will be discretized by finite elements, see, for example, Braess, 2013. We denote the discretized shape by X and will write $J(X, U(X))$ for the discretized objective function with $U(X)$ the solution of the discretized state equation.

4.1 Calculating Shape Sensitivities using Adjoint Equations

In the following chapters we consider shape optimization problems and want to numerically solve them using gradient based methods and finite element analysis. Therefore, we need to calculate the shape gradients of the objective functions. This will be done by a "first discretize then adjoin" approach. Many optimization methods use shape gradients and calculate them by the adjoint approach, see, for example, Conti et al., 2009; Delfour and Zolésio, 2011; Eppler, 2017; Eppler, Harbrecht, and Schneider, 2007; Laurain and Sturm, 2016; Schulz, 2014; Schulz, Siebenborn, and Welker, 2016. We shortly explain the adjoint approach used in this thesis based on Gottschalk and Saadi, 2019. See also Giles and Pierce, 2000 for an introduction to this topic.

Consider the discretized objective function $J(X, U(X))$ of the discretized shape X with $U(X)$ being the solution of the discretized state equation. We want to calculate

the total derivative of J with respect to X:

$$\frac{\mathrm{d}J(X,U(X))}{\mathrm{d}X} = \frac{\partial J(X,U(X))}{\partial X} + \frac{\partial J(X,U(X))}{\partial U}\frac{\partial U(X)}{\partial X}. \tag{4.2}$$

The calculation of the partial derivative $\frac{\partial U(X)}{\partial X}$ is generally very time-consuming since X has the dimension equal to the degrees of freedom resulting from the discretization, which can be huge. Therefore, it is in general not practical. Instead, we will use the adjoint equation to calculate $\frac{\mathrm{d}J(X,U(X))}{\mathrm{d}X}$.

First we have a look at the discretized state equation in weak form. For simplicity, we assume that it is given by

$$B(X)U(X) = F(X) \tag{4.3}$$

with $B(X)$ the corresponding stiffness matrix and $F(X)$ the corresponding right hand side. After differentiating (4.3), we can solve for $\frac{\partial U(X)}{\partial X}$:

$$\begin{aligned} & \frac{\partial \left(B(X)U(X)\right)}{\partial X} = \frac{\partial F(X)}{\partial X} \\ \iff \quad & \frac{\partial B(X)}{\partial X}U(X) + B(X)\frac{\partial U(X)}{\partial X} = \frac{\partial F(X)}{\partial X} \\ \iff \quad & \frac{\partial U(X)}{\partial X} = B(X)^{-1}\left(\frac{\partial F(X)}{\partial X} - \frac{\partial B(X)}{\partial X}U(X)\right). \end{aligned} \tag{4.4}$$

Consider now the Lagrangian formulation for the discretized shape optimization problem, see, for example, Tröltzsch, 2009,

$$\mathscr{L}(X,U,\Lambda) = J(X,U) - \Lambda^{\top}(B(X)U - F(X)) \tag{4.5}$$

with Λ the Lagrange multiplicator, also called adjoint state. Note that only the PDE constraint of (4.1) is eliminated by the Lagrange formulation. The condition that the shape is feasible will be explicitly considered. We have

$$\begin{aligned} \frac{\partial \mathscr{L}(X,U,\Lambda)}{\partial \Lambda} = B(X)U - F(X) &= 0 \\ \iff B(X)U &= F(X), \end{aligned}$$

which is again the state equation.

The adjoint equation is given by (4.6)

$$\frac{\partial \mathscr{L}(X,U,\Lambda)}{\partial U} = \frac{\partial J(X,U)}{\partial U} - \Lambda^\top B(X) = 0$$
$$\iff B(X)\Lambda = \frac{\partial J(X,U)}{\partial U} \tag{4.6}$$

since the stiffness matrix $B(X)$ is symmetric, see Gottschalk and Saadi, 2019. Last we have

$$\frac{\partial \mathscr{L}(X,U,\Lambda)}{\partial X} = \frac{\partial J(X,U(X))}{\partial X} + \Lambda^\top \left(\frac{\partial B(X)}{\partial X} U - \frac{\partial F(X)}{\partial X} \right). \tag{4.7}$$

Inserting (4.4) and (4.6) in (4.2) and comparing to (4.7) we have

$$\frac{\partial \mathscr{L}(X,U,\Lambda)}{\partial X} = \frac{\mathrm{d}J(X,U(X))}{\mathrm{d}X}. \tag{4.8}$$

For more numerical details we refer, for example, to Gottschalk and Saadi, 2019; Hahn, 2021; Saadi, 2021. Later in this thesis we will use shape gradients that are efficiently calculated with adjoint equations. For the probability of failure objectives used in Chapters 5 and 6 we use an implementation of Camilla Hahn, see Hahn, 2021. For the aerodynamic objectives used in Chapter 6 we use the turbo machinery simulation suite, see *TRACE User Guide* 2019.

4.2 Multiobjective Formulation

Shape optimization problems in mechanical engineering often have more than one objective that can be considered. After choosing the material of the component and defining the use cases, a design of the component is searched, that optimizes the desired objectives like functionality, mechanical integrity or cost. In the case of conflicting goals, we then have more than one objective function, which leads us to multiobjective shape optimization. In Chapter 2 we give a short introduction to general multiobjective optimization problems.

In the next chapters we will consider two different biobjective shape optimization problems in engineering applications. In the first application we consider a ceramic rod under tensile load with the two conflicting objectives *volume* and *probability of failure (PoF)*, see Chapter 5. The second application deals with a multi-physical implementation of a gas turbine blade, more specifically, the profile of a blade with

respect to the two objectives *efficiency* and *low-cycle fatigue (LCF)*, see Chapter 6.

A common notation for objective functions in shape optimization problems is the letter J. Therefore, we will adapt this notation and, unlike in Chapter 2, denote the vector-valued objective function by $J\colon \mathcal{O}^{\mathrm{ad}} \to \mathbb{R}^p$ with the individual objective functions $J_1, \ldots, J_p$.

In the following we will also consider individual objectives that does not depend on a PDE (see Chapter 5) and individuals objectives that depend on different PDEs (see Chapter 6). For simplicity, we shortly write $J_k(\Omega)$ for $J_k(\Omega, u(\Omega))$ in the case that an objective J_k is constrained by a PDE since the PDE solution $u(\Omega)$ is uniquely defined by Ω.

For the multiobjective shape optimization problem we then have

$$\begin{aligned} \min \quad & J(\Omega) \coloneqq (J_1(\Omega), \ldots, J_p(\Omega))^\top \\ \text{s. t.} \quad & \Omega \in \mathcal{O}^{\mathrm{ad}}, \\ & \text{state equation(s) are fulfilled.} \end{aligned} \tag{4.9}$$

Here, the feasible solutions are shapes that we denote by $\Omega \subset \mathbb{R}^d$ with dimension $d \in \{2, 3\}$ fixed. The set of admissible shapes will by denoted by $\mathcal{O}^{\mathrm{ad}} (= \mathcal{X})$. For example, technical limitations can be taken into account when defining $\mathcal{O}^{\mathrm{ad}}$.

Existence results for Pareto optimal shapes under certain assumptions are considered, for example, in Doganay et al., 2020; Gottschalk and Reese, 2021; Haslinger and Mäkinen, 2003, but leave space for further research.

Chapter 5

Ceramic Components under Tensile Load

This chapter is already published in Doganay et al., 2020; Schultes et al., 2021. We consider a shape optimization problem concerning reliability and cost of ceramic components under a one-time application of tensile load. The objective concerning reliability of ceramic components under external loads is modeled as a Weibull-type formulation of the probability of failure. These external loads are described by a linear elasticity PDE. Since this reliability objective is introduced in detail in Bolten, Gottschalk, and S. Schmitz, 2015, the most important points will be summarized here. For the objective cost we consider the volume of the component. The numerical implementation of the optimization problem is done by finite element methods and gradient based optimization algorithms. The shape gradients will be calculated by adjoint equations. To solve the discretized shape optimization problem and analyze the trade-off between the two objectives we implemented two different scalarization methods, the weighted sum method and the hypervolume method.

We start with the problem formulation. In Section 5.1 we define the feasible shapes and describe the linear elasticity PDE. Then we introduce the two objectives in Section 5.2. The formulation of the biobjective shape optimization problem is written down in Section 5.3. Afterwards, we turn to the two different numerical implementations. In Section 5.4 we present the first implementation. We implemented the weighted sum method using the programming language R where shapes are fitted by B-splines. In Section 5.5 the we consider another implementation using Python.

There we use a finite element discretization based on structured meshes. This allows us to use a parameter-free representation of the shapes. Based on this discretization we implemented the weighted sum method as well as the hypervolume method.

As previously mentioned, the optimization problem and the numerical results considered in this chapter have already been published in Doganay et al., 2020 and Schultes et al., 2021. The following Sections 5.1, 5.2 and 5.4 are taken from Doganay et al., 2020. Section 5.5 is published in Schultes et al., 2021. We have adapted the notation to the other chapters.

5.1 Setting

In this section we have look at the shape of the component that we want to optimize and define the set of admissible shapes, see Section 5.1.1. In addition, we consider the partial differential equation describing the behavior of the component under tensile load in Section 5.1.2. Based on this setting we will formulate the objective functions in Section 5.2.

5.1.1 Admissible Shapes

We follow the description from Bolten, Gottschalk, Hahn, et al., 2019; Bolten, Gottschalk, and S. Schmitz, 2015 and consider a compact body (also referred to as component or shape) $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, with Lipschitz boundary that is filled with ceramic material. Furthermore, we assume that the boundary $\partial\Omega$ of Ω is subdivided into three parts with nonempty relative interior,

$$\partial\Omega = \mathrm{cl}(\partial\Omega_D) \cup \mathrm{cl}(\partial\Omega_{N_{\mathrm{fixed}}}) \cup \mathrm{cl}(\partial\Omega_{N_{\mathrm{free}}}).$$

The part $\partial\Omega_D$ describes the part of the boundary where the Dirichlet boundary condition holds, $\partial\Omega_{N_{\mathrm{fixed}}}$ the part where surface forces may act on and $\partial\Omega_{N_{\mathrm{free}}}$ the part of the boundary that can be modified in an optimization approach. It is assumed to be force free for technical reasons Bolten, Gottschalk, and S. Schmitz, 2015.

Since all feasible shapes have to coincide in Ω_D and in $\Omega_{N_{\mathrm{fixed}}}$, it is natural to restrict the analysis to subsets of a sufficiently large bounded open set $\widehat{\Omega} \subset \mathbb{R}^d$ that satisfies $\partial\Omega_D \subseteq \partial\widehat{\Omega}$ and $\partial\Omega_{N_{\mathrm{fixed}}} \subseteq \partial\widehat{\Omega}$ (see Figure 5.1). We additionally assume that $\widehat{\Omega}$ satisfies the *cone property* for a given angle $\theta \in (0, \pi/2)$ and radii $r, l > 0$, $r \leq l/2$,

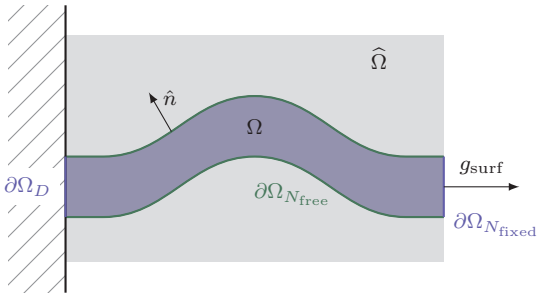


Figure 5.1: Illustration of Ω and its boundary components.

i.e.,

$$\forall x \in \partial\widehat{\Omega}\ \exists \zeta_x \in \mathbb{R}^d, \|\zeta_x\| = 1 \text{ s.t. } y + C(\zeta_x, \theta, l) \subset \widehat{\Omega}\ \forall y \in B(x, r) \cap \widehat{\Omega},$$

where $C(\zeta_x, \theta, l) \coloneqq \left\{c \in \mathbb{R}^d \;\middle|\; \|c\| < l,\ c^\top \zeta_x > \|c\| \cos(\theta)\right\}$ is a truncated circular cone oriented along ζ_x with height l and opening angle 2θ, and $B(x, r) \subset \mathbb{R}^d$ is an open ball of radius r centered at x. Now the set of *admissible shapes* $\mathcal{O}^{\text{ad}} \subset \mathcal{P}(\mathbb{R}^d)$ can be defined as

$$\mathcal{O}^{\text{ad}} \coloneqq \left\{\Omega \subseteq \widehat{\Omega} \;\middle|\; \partial\Omega_D \subseteq \partial\Omega,\ \partial\Omega_{N_{\text{fixed}}} \subseteq \partial\Omega,\ \widehat{\Omega} \text{ and } \Omega \text{ satisfy the cone property}\right\}. \tag{5.1}$$

5.1.2 Linear Elasticity PDE

Ceramic components behave according to the linear elasticity theory (Munz and Fett, 2001). The state equation can be described as an elliptic partial differential equation, see, e.g., Braess, 2013. More precisely, we get the state equation which describes the reaction of the ceramic component to external forces as a partial differential equation:

$$\begin{aligned} -\text{div}(\sigma(u(x))) &= f_{\text{vol}}(x) && \text{for } x \in \Omega \\ u(x) &= 0 && \text{for } x \in \partial\Omega_D \\ \sigma(u(x))\hat{n}(x) &= g_{\text{surf}}(x) && \text{for } x \in \partial\Omega_{N_{\text{fixed}}} \\ \sigma(u(x))\hat{n}(x) &= 0 && \text{for } x \in \partial\Omega_{N_{\text{free}}} \end{aligned} \tag{5.2}$$

Here, $\hat{n}(x)$ is the outward pointing normal at $x \in \partial\Omega$, which is defined almost everywhere on $\partial\Omega$ given that $\partial\Omega$ is piecewise differentiable. Furthermore, let $f_{\text{vol}} \in L^2(\Omega, \mathbb{R}^d)$ be the volume forces and $g_{\text{surf}} \in L^2(\partial\Omega_{N_{\text{fixed}}}, \mathbb{R}^d)$ the forces acting on the surface $\partial\Omega_{N_{\text{fixed}}}$, e.g., the tensile load. The displacement caused by the acting forces is given by $u \in H^1(\Omega, \mathbb{R}^d)$, where $H^1(\Omega, \mathbb{R}^d)$ is the Sobolov space of $L^2(\Omega, \mathbb{R}^d)$-functions

with weak derivatives in $L^2(\Omega, \mathbb{R}^{d\times d})$. The linear strain tensor $\varepsilon \in L^2(\Omega, \mathbb{R}^{d\times d})$ is given by $\varepsilon(u(x)) \coloneqq \frac{1}{2}(Du(x) + (Du(x))^\top)$, where Du is the Jacobi matrix of u. It follows for the stress tensor $\sigma \in L^2(\Omega, \mathbb{R}^{d\times d})$ that $\sigma(u(x)) = \lambda \operatorname{tr}(\varepsilon(u(x)))I + 2\mu\varepsilon(u(x))$, where $\lambda, \mu > 0$ are the Lamé constants derived from Young's modulus E and Poisson's ratio ν as $\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}$ and $\mu = \frac{E}{2(1+\nu)}$.

From a numerical perspective, a variational formulation of the state equation (5.2) is usually preferred, see, e.g., Bolten, Gottschalk, Hahn, et al., 2019; Bolten, Gottschalk, and S. Schmitz, 2015. This still guarantees a unique weak solution u, see Duran and Muschietti, 2004. Thus, u is uniquely defined by the shape Ω (Duran and Muschietti, 2004), and we will equivalently write $\sigma(Du(x)) \coloneqq \sigma(u(x))$ for $x \in \Omega$ to highlight that σ depends on the Jacobi matrix of u.

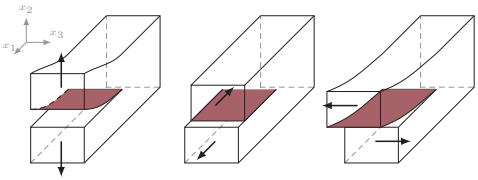
5.2 Objective Functions

5.2.1 Probability of Failure

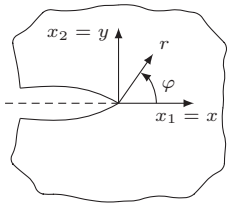
The primary objective function, the mechanical integrity of the ceramic component, is modeled based on the probability of failure analogous to Bolten, Gottschalk, Hahn, et al., 2019; Bolten, Gottschalk, and S. Schmitz, 2015; Brückner-Foit et al., 1997; Weibull, 1939. For the sake of completeness this is briefly summarized in the following.

We want to optimize the reliability of a ceramic body Ω, i.e., its survival probability, by minimizing its probability of failure under tensile load. In that sense failure means that the ceramic body breaks under the tensile load due to cracks. Such cracks occur as a result of small faults in the material caused by the sintering process. To understand the mechanics of cracks, three types of crack opening are considered, see Gross and Seeling, 2006 and Figure 5.2a for an illustration. They are referred to as *Modes I, II* and *III*, respectively, and relate to different loads. Note that in the two-dimensional case, only Modes I and II can occur. We refer to Gross and Seeling, 2006 for a detailed introduction into this topic.

The stresses and strains close to a crack are represented by the *crack-tip field* which depends on the respective crack opening modes. It is described locally by a two-dimensional model, see Figure 5.2b for an illustration. With $K_\mathrm{I}, K_\mathrm{II}$ and K_III being the *stress-intensity factors* (also called *K-factors*) corresponding to Modes I, II, and III, respectively, one can describe the crack-tip field σ locally according to linear



(a) Modes I, II and III (from left to right).



(b) r-φ coordinate system at the tip of the crack.

Figure 5.2: Crack opening modes and two-dimensional model for the crack-tip field according to Bolten, Gottschalk, Hahn, et al., 2019; Bolten, Gottschalk, and S. Schmitz, 2015; Gross and Seeling, 2006.

fracture mechanics as

$$\sigma(x) = \sigma(r,\phi) = \frac{1}{\sqrt{2\pi r}}\Big\{K_{\mathrm{I}}\tilde{\sigma}^{\mathrm{I}}(\phi) + K_{\mathrm{II}}\tilde{\sigma}^{\mathrm{II}}(\phi) + K_{\mathrm{III}}\tilde{\sigma}^{\mathrm{III}}(\phi)\Big\} + R(r,\phi). \tag{5.3}$$

Here, r is the distance to the crack tip, and ϕ the angle w.r.t. the x_1-axis (aligned with the crack plane), see Figure 5.2b. The functions $\tilde{\sigma}^{\mathrm{I,II,III}}(\phi)$ are known functions of the angle ϕ, see again Gross and Seeling, 2006, and $R(r,\phi)$ is a regular function of the considered position in $x \in \Omega$ that is independent of the crack. Note that in the two-dimensional case, Mode III is omitted from (5.3) since it does not exist. Moreover, experimental evidence has shown that Mode I, which relates to tensile and compressive load, is the most relevant for the failure of ceramic structures Brückner-Foit et al., 1997, see Gross and Seeling, 2006 for approaches for multi-mode failure. We will thus focus on K_{I} in the following as the driving parameter for crack development under tensile load.

In order to evaluate K_{I} analogous to Bolten, Gottschalk, and S. Schmitz, 2015, we adopt the concept of equivalent circular discs to represent different crack shapes and crack sizes, and hence assume that the cracks are *penny shaped*. Then a particular crack can be identified by its configuration

$$(x,a,n) \in \mathcal{C} \coloneqq \Omega \times (0,\infty) \times S^{d-1},$$

where $x \in \Omega$ is its location, $a \in (0,\infty)$ its radius, and $n \in S^{d-1}$ its orientation (S^{d-1} denotes the unit sphere in $\mathbb{R}^d$). $\mathcal{C}$ is called the *crack configuration space*. Given a crack $(x,a,n) \in \mathcal{C}$, K_{I} can be computed as a function of the radius a and of the tensile load

$\sigma_n(Du(x))$ in the normal direction n of the stress plane at the crack location x as

$$K_{\mathrm{I}} = K_{\mathrm{I}}(a, \sigma_n(Du(x))) = \frac{2}{\pi}\sigma_n(Du(x))\sqrt{\pi a}, \tag{5.4}$$

see, e.g., Table 4.1 in Gross and Seeling, 2006. Following Bolten, Gottschalk, and S. Schmitz, 2015 we set

$$\sigma_n(Du(x)) \coloneqq \max\{n^\top \sigma(Du(x))\, n\,,\, 0\}.$$

Note that negative values of $\sigma_n(x)$ correspond to compressive loads which can be ignored in the analysis of crack development, see Figure 5.2a above.

A crack $(x, a, n) \in \mathcal{C}$ becomes critical, i.e., a fracture occurs and the material fails, if K_{I} exceeds a material-specific critical value $K_{\mathrm{I}c}$ (the *ultimate tensile strength* of the material). Note that (5.4) implies that all cracks with radius

$$a > a_c \coloneqq \frac{\pi}{4}\left(\frac{K_{\mathrm{I}c}}{\sigma_n(Du(x))}\right)^2 \tag{5.5}$$

are critical. We denote the set of *critical configurations* by

$$A_c \coloneqq A_c(\Omega, Du) = \{(x, a, n) \in \mathcal{C} \mid K_{\mathrm{I}}(a, \sigma_n(Du(x))) > K_{\mathrm{I}c}\}$$

and want to minimize the probability of finding a crack with configuration in A_c.

Following Bolten, Gottschalk, Hahn, et al., 2019; Bolten, Gottschalk, and S. Schmitz, 2015, we assume that the parameters (x, a, n) are random (i.e., they are not deterministically given by the sintering process), that the cracks are statistically homogeneously distributed in Ω, and that their orientations are isotropic. Let $A \subseteq \mathcal{C}$ be a measurable subset of the configuration space. Then under quite general assumptions the random number $N(A)$ of cracks in A is Poisson distributed (see Kallenberg, 1983; Watanabe, 1964), and hence $N(A)$ is a Poisson point process. It follows that $P(N(A) = k) = e^{-\upsilon(A)}\frac{\upsilon(A)^k}{k!} \sim Po(\upsilon(A))$, where υ is the (Radon) *intensity measure* of the process. Recall that a component fails if $N(A_c) > 0$. Given a displacement field $u \in H^1(\Omega, \mathbb{R}^d)$, we can now write the survival probability of the component Ω as

$$p_s(\Omega|Du) = P(N(A_c(\Omega, Du)) = 0) = \exp\{-\upsilon(A_c(\Omega, Du))\}.$$

Hence, to maximize the survival probability of a component Ω we need to minimize the intensity measure υ. Since only cracks (x, a, n) with radius $a > a_c$ need to be

considered (c.f. (5.5) above), Bolten, Gottschalk, Hahn, et al., 2019; Bolten, Gottschalk, and S. Schmitz, 2015 determine the intensity measure as

$$\upsilon(A_c(\Omega, Du)) = \frac{\Gamma(\frac{d}{2})}{2\pi^{\frac{d}{2}}} \int_{\Omega} \int_{S^{d-1}} \int_{a_c}^{\infty} \mathrm{d}\upsilon_a(a)\,\mathrm{d}n\,\mathrm{d}x$$

with $\mathrm{d}x$ the Lebesgue measure on $\mathbb{R}^d$, $\mathrm{d}n$ the surface measure on S^{d-1}, and $\mathrm{d}\upsilon_a(a) = c \cdot a^{-\tilde{m}}\mathrm{d}a$ being a positive Radon measure modeling the occurrence of cracks of radius a in Ω ($c > 0$ and $\tilde{m} \geq \frac{3}{2}$ are positive constants). Note that for $d = 3$ the Γ-function takes the value $\Gamma(\frac{3}{2}) = \frac{\sqrt{\pi}}{2}$ and for $d = 2$ we obtain $\Gamma(1) = 1$. With $m \coloneqq 2(\tilde{m} - 1) \geq 1$ and using again (5.5) the inner integral can be evaluated, yielding

$$\upsilon(A_c(\Omega, Du)) = \frac{\Gamma(\frac{d}{2})}{2\pi^{\frac{d}{2}}} \int_{\Omega} \int_{S^{d-1}} \left(\frac{\sigma_n(Du(x))}{\sigma_0}\right)^m \mathrm{d}n\,\mathrm{d}x,$$

where σ_0 is an appropriately chosen positive constant. As highlighted in Bolten, Gottschalk, Hahn, et al., 2019; Bolten, Gottschalk, and S. Schmitz, 2015, this is in accordance with the statistical model introduced by Weibull Weibull, 1939. In this context, the parameter m is referred to as *Weibull module* and typically assumes values between 5 and 25.

Summarizing the discussion above, we define our primary objective function $J_1 \colon \mathcal{O}^{\mathrm{ad}} \to \mathbb{R}$ as

$$J_1(\Omega) \coloneqq \upsilon(A_c(\Omega, Du)) \tag{5.6}$$

and refer to it as *intensity measure*, modeling the probability of failure (PoF) of the component Ω. Recall that $u(\Omega)$ is uniquely defined by Ω and thus $J_1(\Omega)$ is completely defined by the shape Ω (given fixed boundary conditions f, g).

5.2.2 Material Consumption

Improving the intensity measure J_1 of a ceramic component (and hence its PoF) usually comes at the price of an increased material consumption, which is directly correlated with the cost of the component. In order to avoid excessively expensive solutions, classical approaches thus set a predetermined bound on the allowable volume of the shape Ω (see, e.g., Bolten, Gottschalk, Hahn, et al., 2019; Bolten, Gottschalk, and S. Schmitz, 2015). We follow a more general approach in this manuscript and

interpret the volume (and hence the cost) of the component as an equitable second objective function. This facilitates, in particular, the analysis of the trade-off between these two criteria and supports the engineer in finding a preferable design. We thus define $J_2 \colon \mathcal{O}^{\mathrm{ad}} \to \mathbb{R}$ as the *volume* of a shape $\Omega \in \mathcal{O}^{\mathrm{ad}}$ given by

$$\mathrm{V}(\Omega) = J_2(\Omega) \coloneqq \int_{\Omega} \mathrm{d}x. \tag{5.7}$$

5.3 Biobjective Shape Optimization Problem Formulation

The *biobjective PDE constrained shape optimization problem* w.r.t. reliability and cost of a ceramic component, is given by

$$\begin{aligned} \min \quad & J_1(\Omega) = \upsilon(A_c(\Omega, Du)) && (5.6)\\ \min \quad & J_2(\Omega) = \mathrm{V}(\Omega) && (5.7)\\ \text{s. t.} \quad & u \in H^1(\Omega, \mathbb{R}^d) \text{ solves the state equation (5.2),} \\ & \Omega \in \mathcal{O}^{\mathrm{ad}}. \end{aligned} \tag{5.8}$$

As described in Section 5.1.1, $\mathcal{O}^{\mathrm{ad}}$ is the set of admissible shapes. The objective function $\upsilon(A_c(\Omega, Du))$ is an intensity measure modeling the probability of failure as described in Subsection 5.2.1 and the objective function $\mathrm{V}(\Omega)$ is the volume of the component representing its cost, see Subsection 5.2.2.

5.3.1 Existence of Pareto Optimal Shapes

In order to prove the existence of Pareto optimal shapes, we consider the weighted sum scalarization of problem (5.8) in which, given a weight $\omega \in (0, 1)$, the two objective functions are combined into one single weighted sum objective:

$$\begin{aligned} & \min_{\Omega \in \mathcal{O}^{\mathrm{ad}}} J_\omega(\Omega) \coloneqq \omega J_1(\Omega) + (1-\omega) J_2(\Omega) \\ & \text{s. t. } u \in H^1(\Omega, \mathbb{R}^d) \text{ solves the state equation (5.2),} \\ & \qquad \Omega \in \mathcal{O}^{\mathrm{ad}}. \end{aligned} \tag{5.9}$$

It is a well-known fact that every optimal solution of problem (5.9) is Pareto optimal for problem (5.8), see, e.g., Ehrgott, 2005 and Section 2.2.1. We denote set of all Pareto optimal shapes with $\mathcal{O}^{\mathrm{ad}}_{\mathrm{E}}$.

Theorem 10. *If the crack size measure has the non decreasing stress hazard property (see Bolten, Gottschalk, and S. Schmitz, 2015 for a formal definition), then the set $\mathcal{O}_E^{ad}$ is non-empty.*

Proof. Suppose that $\omega \in (0,1)$ is chosen arbitrarily, but fixed. Then the weighted sum objective can be evaluated as

$$\begin{aligned} J_\omega(\Omega) &= \omega \left(\frac{\Gamma(\frac{d}{2})}{2\pi^{\frac{d}{2}}} \int_\Omega \int_{S^{d-1}} \left(\frac{\sigma_n(Du(x))}{\sigma_0} \right)^m \mathrm{d}n \, \mathrm{d}x \right) + (1-\omega) \int_\Omega \mathrm{d}x \\ &= \omega \frac{\Gamma(\frac{d}{2})}{2\pi^{\frac{d}{2}}} \int_\Omega \int_{S^{d-1}} \left(\frac{\sigma_n(Du(x))}{\sigma_0} \right)^m \mathrm{d}n \; + \; \underbrace{\frac{2\pi^{\frac{d}{2}}(1-\omega)}{\Gamma(\frac{d}{2})\omega}}_{\text{constant}} \mathrm{d}x. \end{aligned}$$

Thus, the incorporation of J_2 into the scalarized objective function corresponds to the addition of a constant term in the shape integral of J_1. This does not affect the convergence analysis of Bolten, Gottschalk, and S. Schmitz, 2015, which is based on convexity of the integrand in Du, see Chenais, 1975; Fujii, 1988. We can conclude that the weighted sum scalarization has an optimal solution for every $\omega \in (0,1)$. Since every such solution is Pareto optimal for (5.8), the result follows. □

5.4 Implementation with B-Splines

The first implementation considered in this thesis is the implementation of the weighted sum method, see Section 2.2.1, in the programming language R. Here we chose a parameter based representation for the shapes. The shapes are described by the meanline and thickness and are fitted with B-splines. This section is taken from Doganay et al., 2020.

To actually compute locally Pareto optimal shapes, we adopt the finite element discretization implemented in Bolten, Gottschalk, Hahn, et al., 2019 for two-dimensional instances (i. e., $p = 2$). In this implementation, the shapes $\Omega \in \mathcal{O}^{\mathrm{ad}}$, the state equation (5.2), the objective functions J_1 and J_2 and their gradients are discretized. Standard Lagrangian finite elements are used for the discretization of the state equation (5.2), and all integrals are calculated using numerical quadrature. The discretized shape gradients are obtained by an adjoint approach to reduce computational costs. We refer to Bolten, Gottschalk, Hahn, et al., 2019 for a detailed description.

5.4.1 Geometry Definition and Finite Element Mesh

The two-dimensional shapes $\Omega \in \mathcal{O}^{\mathrm{ad}} \subset \mathcal{P}(\mathbb{R}^2)$ are discretized by an $n_x \times n_y$ mesh $X := X^\Omega = (X_{ij}^\Omega)_{n_x \times n_y}$ (we write $X_{ij} := X_{ij}^\Omega \in \mathbb{R}^2$ for short) using triangles, with $n_x, n_y \in \mathbb{N}$ being the number of grid points in x and y direction, respectively. Given a shape $\Omega \in \mathcal{O}^{\mathrm{ad}}$ and its discretization X, the objective function values $J_1(X)$ and $J_2(X)$ as well as their gradients $\nabla J_1(X)$ and $\nabla J_2(X)$ are computed using the implementation of Bolten, Gottschalk, Hahn, et al., 2019.

For the optimization process, we fix the x-component of all grid points to equidistant values $x_1, \ldots, x_{n_x}$, and we only consider the y-components of those grid points that define the boundary of the shape to avoid deformation of the inner mesh structure. Note that this reformulation reduces the number of optimization variables from $2n_xn_y$ to $2n_x$. As a consequence, feasible shapes can alternatively be represented by a *shape parameter* ϱ containing, for every relevant x-coordinate, the y-coordinate of the *meanline* $\varrho_i^{\mathrm{ml}} \in \mathbb{R}$ of the shape, and the *thickness* $\varrho_i^{\mathrm{th}} \in \mathbb{R}_>$ of the shape, $i = 1, \ldots, n_x$. Given a feasible shape represented by $\varrho := \left(\varrho^{\mathrm{ml}}, \varrho^{\mathrm{th}}\right) \in \mathbb{R}^{2n_x}$ with $\varrho^{\mathrm{th}} \in \mathbb{R}_>^{n_x}$, an associated mesh representation X can be obtained using

$$X_{i,j} := \left(x_i \ , \ \varrho_i^{\mathrm{ml}} + \frac{\varrho_i^{\mathrm{th}}}{n_y - 1} \left(j - \frac{n_y + 1}{2} \right) \right) \in \mathbb{R}^2, \qquad i = 1, \ldots, n_x, \ j = 1, \ldots, n_y. \tag{5.10}$$

To further reduce the computational burden and to obtain smoother shapes, the shape parameters $\varrho^{\mathrm{ml}} \in \mathbb{R}^{n_x}$ and $\varrho^{\mathrm{th}} \in \mathbb{R}_>^{n_x}$ are modeled using *B-splines*. Let $n_B \in \mathbb{N}$, with $n_B < n_x$, be the number of B-spline basis functions, and let $\{\vartheta_j : \mathbb{R} \to \mathbb{R}_\geq, \ j = 1, \ldots, n_B\}$ be a B-spline basis (see, e.g., Les Piegl, 2000). Feasible shapes are then represented by B-spline coefficients $\gamma := \left(\gamma^{\mathrm{ml}}, \gamma^{\mathrm{th}}\right) \in \mathbb{R}^{2n_B}$. The corresponding meanline and thickness values can be computed using the auxiliary functions

$$\hat{\varrho}^{\mathrm{ml}}(x) := \sum_{j=1}^{n_B} \gamma_j^{\mathrm{ml}} \, \vartheta_j(x) \quad \text{and} \quad \hat{\varrho}^{\mathrm{th}}(x) := \sum_{j=1}^{n_B} \gamma_j^{\mathrm{th}} \, \vartheta_j(x), \qquad x \in \mathbb{R}.$$

These auxiliary meanline and thickness functions are then evaluated at the fixed x-coordinates of the grid points which yields

$$\varrho_i^{\mathrm{ml}} := \hat{\varrho}^{\mathrm{ml}}(x_i) \quad \text{and} \quad \varrho_i^{\mathrm{th}} := \hat{\varrho}^{\mathrm{th}}(x_i), \qquad i = 1, \ldots, n_x. \tag{5.11}$$

Using the B-spline coefficients $\gamma = \left(\gamma^{\mathrm{ml}}, \gamma^{\mathrm{th}}\right) \in \mathbb{R}^{2n_B}$ as optimization variables yields a further reduction of the number of variables to $2n_B$. Moreover, the B-spline

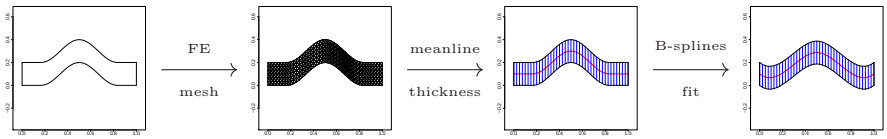


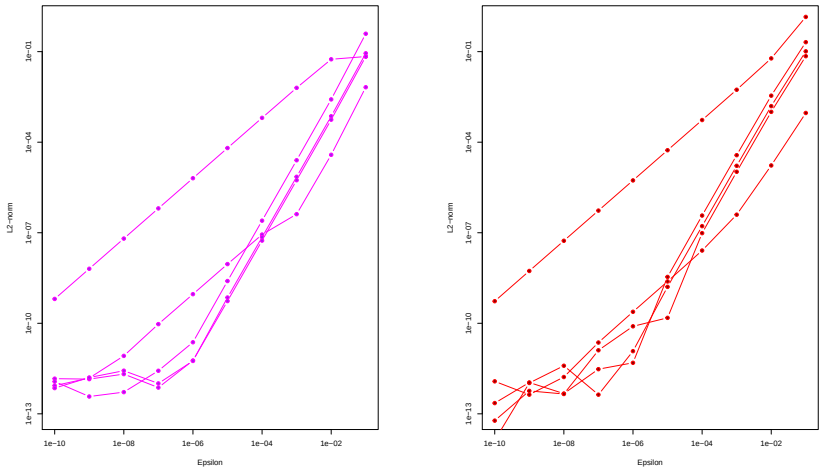
Figure 5.3: Transformation: mesh $X \to$ meanline/thickness $\varrho \to$ B-spline fit γ.

representation leads to an implicit regularization and smoothing of the represented shapes. In the following, we denote the set of *feasible shape parametrizations* by $\Gamma \subseteq \{(\gamma^{\mathrm{ml}}, \gamma^{\mathrm{th}}) \in \mathbb{R}^{2n_B}\}$. The transformation from the mesh to the B-spline fit is visualized in Figure 5.3.

To evaluate the objective functions $J_j(\gamma)$ and their gradients $\nabla J_j(\gamma) = \partial J_j / \partial \gamma$, $j = 1, 2$, w.r.t. the new parametrization of shapes based on B-spline parameters γ, while still using the implementation of Bolten, Gottschalk, Hahn, et al., 2019, we compute an associated grid X using a two step transformation. First, the fixed B-spline basis is utilized to construct the auxiliary functions of meanline and thickness, the evaluation of which (via (5.11)) then generates the shape parameters for the next step of the grid computation (5.10). While the resulting objective function values can be used immediately in the optimization process, the gradients computed w.r.t. the grid X need to be translated to the space of B-spline coefficients, i. e.,

$$\frac{\partial J_j}{\partial \gamma^{\mathrm{ml}}} = \frac{\partial J_j}{\partial X} \frac{\partial X}{\partial \varrho^{\mathrm{ml}}} \frac{\partial \varrho^{\mathrm{ml}}}{\partial \gamma^{\mathrm{ml}}} \quad \text{and} \quad \frac{\partial J_j}{\partial \gamma^{\mathrm{th}}} = \frac{\partial J_j}{\partial X} \frac{\partial X}{\partial \varrho^{\mathrm{th}}} \frac{\partial \varrho^{\mathrm{th}}}{\partial \gamma^{\mathrm{th}}}, \qquad j = 1, 2. \tag{5.12}$$

The numerical computation of gradients of J_j, $j = 1, 2$, w.r.t. a B-spline representation γ of a feasible shape Ω is thus based on a two-step projection of γ onto the original grid X. The thus computed gradients of J_1 (the intensity measure) were validated, using finite differences, at the sample shape shown in Figure 5.5a. The validation is based on a grid $(X_{ij})_{41\times 7}$, i. e., $n_x = 41$ and $n_y = 7$. Consequently, for the corresponding meanline and thickness representation we have $\varrho = (\varrho^{\mathrm{ml}}, \varrho^{\mathrm{th}}) \in \mathbb{R}^{82}$, where $\varrho^{\mathrm{th}} \in \mathbb{R}^{41}_{>}$. Moreover, we used a B-spline basis with five basis functions, i. e., $n_B = 5$ and $\gamma = (\gamma^{\mathrm{ml}}, \gamma^{\mathrm{th}}) \in \mathbb{R}^{10}$. We computed all ten partial derivatives w.r.t. γ via the respective transformations to the grid representation and compared them with finite differences. The results of this comparison, i. e., the absolute values of the differences between computed derivatives and finite differences, are shown in Figure 5.4a and 5.4b for the meanline and thickness parameters, respectively. The figures indicate in all cases that when the finite differences are evaluated for decreasing



(a) Validation of $\partial J_1/\partial \gamma_i^{\mathrm{ml}}$ (b) Validation of $\partial J_1/\partial \gamma_i^{\mathrm{th}}$

Figure 5.4: Validation of gradients computed according to (5.12) using finite differences. On the x-axis: increment ε used for the finite difference evaluation; on the y-axis: absolute deviation between $\partial J_1/\partial \gamma_i^{\mathrm{ml,th}}$ computed according to (5.12) and the corresponding finite difference, $i = 1, \ldots, 5$, for meanline (left) and thickness (right).

values of the increment ε, then they correspond well to the computed gradients.

5.4.2 Pareto Critical Shapes

Given the parametrization of admissible shapes described in Section 5.4.1, the biobjective optimization problem (5.8) can now be restated as

$$\begin{aligned} &\min_{\gamma \in \Gamma} (J_1(\gamma), J_2(\gamma)) \\ &\text{s.t. } u(X(\gamma)) \text{ solves the discretized state equation (5.2).} \end{aligned} \tag{5.13}$$

Recall that only J_1 depends on the displacement field $u(X)$.

In this work, we aim at the efficient computation of Pareto critical shapes (see (2.11)) that, ideally, approximate the Pareto front. Since derivative information can be obtained for both objective functions, we select solution methods that efficiently utilize this information and that can be adopted such that a meaningful representation of a Pareto critical front is obtained. As two fundamental approaches in this category, a parametrized weighted sum method and a biobjective descent algorithm are chosen

and explained in Sections 5.4.3 and 5.4.4, respectively. Their performance in the context of 2D shape optimization problems is compared in Section 5.4.7.

5.4.3 Weighted Sum Method

Maybe the easiest way to compute a representation of the Pareto front is to iteratively solve weighted sum scalarizations (5.9) with varying weights, see Section 2.2.1. The weighted sum scalarization of problem (5.13) can be restated as

$$\begin{aligned} &\min_{\gamma \in \Gamma} \; J_w(\gamma) := \omega J_1(\gamma) + (1-\omega) J_2(\gamma) \\ &\text{s.t. } u(X(\gamma)) \text{ solves the discretized state equation (5.2),} \end{aligned} \tag{5.14}$$

where $\omega \in (0,1)$ is the *weight* specifying the relative importance of J_1 and J_2, respectively. Recall that every solution of the weighted sum scalarization (5.14) is Pareto optimal for (5.13) Ehrgott, 2005. A disadvantage of the weighted sum method is, however, that only solutions that map to the convex hull $\mathrm{conv}(Z)$ of the image set $Z = f(\Gamma)$ in the objective space can be found, and thus relevant compromise solutions in non-convex areas of the non-dominated front may be missed. Moreover, Das and Dennis, 1997 showed at simple biobjective test instances that evenly distributed weights do in general not lead to well distributed outcome vectors in the objective space. This is particularly problematic if the considered objective function values are of largely different magnitude, which is the case here. In order to obtain solutions that are consistent with the preferences expressed by ω, we thus normalize the objective functions by using appropriate scaling factors $c_1, c_2 > 0$, and replace J_1 and J_2 in (5.14) by $c_1 J_1$ and $c_2 J_2$, respectively.

Despite the difficulties mentioned above, the weighted sum method is usually well-suited to efficiently compute at least a rough approximation of the Pareto front. For this purpose, problem (5.14) is solved iteratively for varying weights (in our case, we choose $\omega \in \{0.2, 0.25, 0.3, \ldots, 0.9\}$ since numerical experiments showed that this yields meaningful trade-offs). Each single-objective optimization problem (5.14) is then individually solved using a classical gradient descent algorithm with stepsizes determined according to the Armijo rule, see, for example, Bazaraa, 2006.

Under appropriate assumptions, the gradient descent algorithm in the inner loop of Algorithm 3 converges to a critical point of (5.14), see, e.g., Bazaraa, 2006. In our implementation, the inner loop is also terminated when a prespecified maximum number of iterations is reached. However, in this case there is no guarantee that the

Data: Choose $\beta \in (0,1)$, $\gamma^{(1)} \in \Gamma$, weights $\omega_1, \ldots, \omega_J \in (0,1)$, and $\varepsilon > 0$.
Result: Set of approximations of Pareto critical solutions $\tilde{\gamma}_1, \ldots, \tilde{\gamma}_J$.
for $j = 1$ **to** J **do**
  Set $\omega = \omega_j$, set $k := 1$, and set $d^{(0)} := -\nabla J_\omega(\gamma^{(1)})$ and $t_0 := 1$;
  while $\|t_{k-1}\, d^{(k-1)}\| > \varepsilon$ **do**
    Compute a search direction $d^{(k)} = -\nabla J_\omega(\gamma^{(k)})$;
    Compute a step length $t_k \in (0,1]$ as

$$\max\left\{ t = \frac{1}{2^\ell} : \ell \in \mathbb{N}_0, \; J_\omega(\gamma^{(k)} + t d^{(k)}) \leq J_\omega(\gamma^{(k)}) + \beta t \nabla J_\omega(\gamma^{(k)})^\top d^{(k)} \right\};$$

    $\gamma^{(k+1)} := \gamma^{(k)} + t_k\, d^{(k)}$ and $k := k + 1$;
  end
  $\tilde{\gamma}_j := \gamma^{(k)}$
end

Algorithm 3: Parametric weighted sum algorithm using gradient descent

final iterate is close to a Pareto critical solution.

Note that a critical point of the weighted sum scalarization (5.14) is necessarily Pareto critical for the biobjective shape optimization problem (5.13), while the converse is not true in general. This has some correspondence to the fact that global optimal solutions of a weighted sum scalarization (5.14) are always Pareto optimal, while non-convex problems may have Pareto optimal solutions that are not optimal for any weighted sum scalarization (5.14), see, e.g., Ehrgott, 2005.

Note also that the search direction $d^{(k)} = -\nabla J_\omega(\gamma^{(k)})$ does not necessarily satisfy $\nabla J_j(\gamma^{(k)})^\top d^{(k)} < 0$, $j = 1, 2$, in all iterations. In other words, one objective function may deteriorate during the optimization process if only the other objective function compensates for this.

5.4.4 Biobjective Descent Algorithm

The biobjective descent algorithm is described in Section 2.2.3 and the implementation of the method for the shape optimization problem considered in this chapter is summarized in Algorithm 4. Towards this end, we omit the constraints implied by the parametric representation of admissible shapes to keep the exposition simple. All constraints will be handled implicitly in the numerical tests described in Section 5.4.7 below.

Data: Choose $\beta \in (0,1)$, $\gamma^{(1)} \in \Gamma$ and $\varepsilon > 0$, set $k := 1$.
Result: Approximation of a Pareto critical solution $\tilde{\gamma} := \gamma^{(k)}$.
Compute $d^{(0)} := d^{(1)}$ as a solution of (2.12) and set $t_0 := 1$;
while $\|t_{k-1}\, d^{(k-1)}\| > \varepsilon$ **do**
  Compute $d^{(k)}$ as a solution of (2.12);
  Compute a step length $t_k \in (0,1]$ as

$$\max\left\{ t = \frac{1}{2^\ell} \;:\; \ell \in \mathbb{N}_0,\; J_j(\gamma^{(k)} + t d^{(k)}) \le J_j(\gamma^{(k)}) + \beta t \nabla J_j(\gamma^{(k)})^\top d^{(k)},\; j = 1,2 \right\};$$

  $\gamma^{(k+1)} := \gamma^{(k)} + t_k\, d^{(k)}$ and $k := k+1$;
end

Algorithm 4: Biobjective descent algorithm according to Fliege and Svaiter, 2000

If J_1 and J_2 are continuously differentiable and $\varepsilon = 0$, then Algorithm 4 converges to a Pareto critical solution Fliege and Svaiter, 2000. A natural stopping condition for practical implementations, motivated by (2.11), is that $\|t_k d^{(k)}\| \le \varepsilon$, with $\varepsilon > 0$ a prespecified small constant.

In practice, we also terminate the algorithm when a prespecified maximum number of iterations is reached. In this case, the final solution has to be used with caution since the optimization procedure has generally not yet converged.

The choice of the search direction using problem (2.12) together with condition (2.13) implies that the iterates of Algorithm 4 satisfy $f(\gamma^{(k+1)}) < f(\gamma^{(k)})$ for all $k = 1, 2, \ldots$. In other words, the objective vector $f(\gamma^{(k+1)})$ in iteration $k+1$ is bounded above by the objective vector $f(\gamma^{(k)})$ of the previous iteration k, i. e., $f(\gamma^{(k+1)}) \in f(\gamma^{(k)}) - \mathbb{R}^2_>$.

Several alternative Pareto critical solutions (and hence trade-off information between them) can be obtained, for example, by varying the starting solution. We follow a different approach in our implementation that is somewhat similar to the weighted sum method, and that is based on the observation that the optimal solution of problem (2.12) (i. e., the direction of steepest biobjective descent) depends on the scaling of the objective functions J_1 and J_2. Thus, Algorithm 4 is executed repeatedly, using different scalings of the objective functions. In our implementation, we use a scaling parameter $s := \bar{\omega} r^{\max} > 0$ and replace J_2 by sJ_2 in the optimization process, where

the parameter $r^{\max} > 0$ is chosen as the largest ratio between partial derivatives of J_1 and J_2, evaluated at the starting solution $\gamma^{(1)}$. Note that the latter aims at the constraints in problem (2.12) in the sense that they should be comparable, i. e., both objective functions should equally contribute to active constraints and thus influence the choice of the search direction. By varying the parameter $\bar{\omega} \in \{0.5, 0.6, \ldots, 2\}$, we implicitly control the run of the gradient descent algorithm and thus obtain different solutions starting from the same initial shape. Note that the volume of the solutions can be expected to increase with larger values of $\bar{\omega}$.

Note also that the resulting parametric version of Algorithm 4 is fundamentally different from the weighted sum method in Algorithm 3 in the way the search directions are chosen and in the way the iterates converge to a Pareto critical solution.

5.4.5 Scalar Products and Gradients in Shape Optimization

The performance of Algorithms 3 and 4 depends largely on the choice of the search direction, which is computed based on the discretized gradients $\nabla J_j(\gamma)$, $j = 1, 2$. In Michor and Mumford, 2006 it is shown that (continuous) shape gradients calculated with respect to the ordinary L^2-scalar product lead to an ill defined notion of the distance of two shapes, as the infimum over all deformation path lengths is zero. There is a modified scalar product suggested given by

$$\langle h, k \rangle_\xi = \int_{\partial\Omega} \langle h, k \rangle_{\mathbb{R}^2} \left(1 + \xi\kappa^2\right) \mathrm{d}A \tag{5.15}$$

and it is shown that this indeed leads to a well defined Riemannian metric on the shape space. Here, h, k are two vector fields in normal direction to the boundary of $\partial\Omega$, $\mathrm{d}A$ is the induced surface measure, κ is the scalar curvature of the surface, and $\xi > 0$ is a regularization parameter. In practice, this corresponds to a transformation of function values on $\partial\Omega$ that, given some function $g : \partial\Omega \to \mathbb{R}^2$, can be described by $g_\xi(x) = \frac{g(x)}{1+\xi\kappa^2(x)}$ for $x \in \partial\Omega$.

Despite discretizing the space of shapes, we also discretize this definition of the gradient in order to obtain stability in the limit of small finite element mesh size and a high number of spline basis elements. We adopt a discretized version of this concept in the numerical implementation of shape gradients for both objectives J_j, $j = 1, 2$. More precisely, a discretized scalar curvature κ is computed at grid points on the boundary $\partial\Omega$, which is represented by a polygonal approximation induced by the shape parameters $\left(\varrho^{\mathrm{ml}}, \varrho^{\mathrm{th}}\right) \in \mathbb{R}^{2n_x}$, $\varrho^{\mathrm{th}} \in \mathbb{R}^{n_x}_{>}$. Since the upper and lower boundary

of the shape Ω may have a different curvature at the same x-coordinate value x_i ($i \in \{1, \ldots, n_x\}$), we have to compute the curvature for upper and lower boundary points separately. For the upper boundary, this is realized by comparing the normals n_i^{u} and n_{i+1}^{u} on two consecutive facets of length l_i^{u} and l_{i+1}^{u}, respectively. Similarly, for the lower boundary we use n_i^{l}, n_{i+1}^{l} and l_i^{l}, l_{i+1}^{l}, and obtain

$$\begin{aligned} \kappa_i^{\mathrm{u}} &:= \kappa^{\mathrm{u}}(x_i) = \frac{2\left\|n_i^{\mathrm{u}} - n_{i+1}^{\mathrm{u}}\right\|_2}{l_i^{\mathrm{u}} + l_{i+1}^{\mathrm{u}}}, \\ \kappa_i^{\mathrm{l}} &:= \kappa^{\mathrm{l}}(x_i) = \frac{2\left\|n_i^{\mathrm{l}} - n_{i+1}^{\mathrm{l}}\right\|_2}{l_i^{\mathrm{l}} + l_{i+1}^{\mathrm{l}}}, \end{aligned} \qquad i = 1, \ldots, n_x - 1. \tag{5.16}$$

The upper and lower boundaries of the shape Ω are reconstructed from the meanline and thickness representation using the linear transformation $\varrho_i^{\mathrm{u}} = \varrho_i^{\mathrm{ml}} + \frac{1}{2}\varrho_i^{\mathrm{th}}$ and $\varrho_i^{\mathrm{l}} = \varrho_i^{\mathrm{ml}} - \frac{1}{2}\varrho_i^{\mathrm{th}}$, $i = 1, \ldots, n_x$. In other words, $\left(\varrho^{\mathrm{u}}, \varrho^{\mathrm{l}}\right) \in \mathbb{R}^{2n_x}$ is obtained from $\left(\varrho^{\mathrm{ml}}, \varrho^{\mathrm{th}}\right) \in \mathbb{R}^{2n_x}$, $\varrho^{\mathrm{th}} \in \mathbb{R}^{n_x}_{>}$, as $\left(\varrho^{\mathrm{u}}, \varrho^{\mathrm{l}}\right) = M\left(\varrho^{\mathrm{ml}}, \varrho^{\mathrm{th}}\right)$, using an appropriate transformation matrix $M \in \mathbb{R}^{2n_x \times 2n_x}$. This leads to a discretized representation of the respective boundaries by points $(x_i, \varrho_i^{\mathrm{u}})$ (upper boundary) and $\left(x_i, \varrho_i^{\mathrm{l}}\right)$ (lower boundary), from which the κ values can be computed according to (5.16).

Now (5.15) can be applied to the gradients of J_j w.r.t. $\left(\varrho^{\mathrm{u}}, \varrho^{\mathrm{l}}\right)$, $j = 1, 2$, by multiplying the respective partial derivatives by

$$d_{\xi,i}^{\mathrm{u}} := \frac{1}{1 + \xi\,(\kappa_i^{\mathrm{u}})^2} \quad \text{and} \quad d_{\xi,i}^{\mathrm{l}} := \frac{1}{1 + \xi\,\left(\kappa_i^{\mathrm{l}}\right)^2}, \quad i = 1, \ldots, n_x.$$

Since we actually need the gradients of J_j w.r.t. $\varrho = \left(\varrho^{\mathrm{ml}}, \varrho^{\mathrm{th}}\right)$, $j = 1, 2$, we additionally have to consider the linear transformation M. Let $D_\xi = (d_{\xi,ij})_{2n_x \times 2n_x} \in \mathbb{R}^{2n_x \times 2n_x}$ be a diagonal matrix with diagonal elements given by

$$d_{\xi,ii} := d_{\xi,i}^{\mathrm{u}}, \quad i = 1, \ldots, n_x \quad \text{and} \quad d_{\xi,ii} := d_{\xi,i-n_x}^{\mathrm{l}}, \quad i = n_x + 1, \ldots, 2n_x,$$

and set $\bar{D}_\xi := M^{-1} D_\xi M$. Then we obtain the curvature adapted B-spline gradients as

$$\left(\frac{\partial J_j}{\partial \gamma}\right)_\xi = \bar{D}_\xi \left(\frac{\partial J_j}{\partial X}\frac{\partial X}{\partial \varrho}\right)\frac{\partial \varrho}{\partial \gamma}, \qquad j = 1, 2. \tag{5.17}$$

Note that for $\xi = 0$ the matrix $\bar{D}_0$ is the identity matrix, and hence the L^2-gradient of J_j w.r.t. γ, $j = 1, 2$, is recovered in this case, c.f. (5.12).

5.4.6 Control of Step Sizes

Large mesh deformations may cause numerical difficulties and thus have to be avoided. We thus limit the step size during the optimization procedure. Recall that the representation of feasible shapes, using meanline and thickness values $\left(\varrho_i^{\mathrm{ml}}, \varrho_i^{\mathrm{th}}\right)$ at fixed x_i coordinates, $i = 1, \ldots, n_x$, implies that grid points can only move vertically. A natural choice for a maximum admissible step in one iteration of the optimization process is thus determined by the thickness of the shape, divided by the number n_y of grid points in y-direction. Since in our case studies the shapes are fixed at the left boundary (i. e., at $x = x_1$) and hence their thickness is constant at x_1, we set

$$\delta^{\max} := 0.8\, \frac{\varrho_1^{\mathrm{th},(1)}}{n_y}$$

i. e., to 80% of the vertical distance between grid points on the left boundary of the initial shape. For a given search direction $d^{(k)} = \left(d^{\mathrm{ml},(k)}, d^{\mathrm{th},(k)}\right) \in \mathbb{R}^{2n_B}$ in iteration k of the optimization algorithms, we check whether

$$\max_{i=1,\ldots,2n_B} \left|d_i^{(k)}\right| \leq \delta^{\max}.$$

Otherwise, $d^{(k)}$ is scaled by a factor $\delta^{\max} / \max_{i=1,\ldots,2n_B} \left|d_i^{(k)}\right|$. Then the step length $t \leq 1$ is computed according to the Armijo rule as indicated in Algorithms 3 and 4.

While $\delta^{\max}$ is derived from the mesh $X^{(1)}$, it still is a meaningful upper bound for a step $d^{(k)}$ in the B-spline representation. Indeed, if $\{\vartheta_j,\, j = 1, \ldots, n_B\}$ is a B-spline basis and $\gamma^{(k)} = \left(\gamma^{\mathrm{ml},(k)}, \gamma^{\mathrm{th},(k)}\right) \in \Gamma$ is the current iterate, then the B-spline basis properties $\sum_{j=1}^{n_B} \vartheta_j(x) = 1$ and $\vartheta_j(x) \geq 0$, $j = 1, \ldots, n_B$ (see, e.g., Les Piegl, 2000) imply that, for all $i = 1, \ldots, n_x$,

$$\begin{aligned}
\left|\varrho_i^{\mathrm{ml},(k+1)} - \varrho_i^{\mathrm{ml},(k)}\right| &= \left|\sum_{j=1}^{n_B} (\gamma_j^{\mathrm{ml},(k)} + d_j^{\mathrm{ml},(k)})\vartheta_j(x_i) - \sum_{j=1}^{n_B} \gamma_j^{\mathrm{ml},(k)} \vartheta_j(x_i)\right| \\
&\leq \sum_{j=1}^{n_B} \left|d_j^{\mathrm{ml},(k)}\right| \, |\vartheta_j(x_i)| \leq \max_{j=1,\ldots,n_B} \left|d_j^{\mathrm{ml},(k)}\right| \sum_{j=1}^{n_B} |\vartheta_j(x_i)| = \max_{j=1,\ldots,n_B} \left|d_j^{\mathrm{ml},(k)}\right|.
\end{aligned}$$

An analogous bound holds for the corresponding thickness parameters. Note that the above inequalities do in general not guarantee that *all* grid points of the corresponding mesh $X^{(k)}$ move by at most 80%, since this also depends on the current shape and the mutual movement of meanline and thickness values. In some situations it may thus be

necessary to adapt this bound to a smaller value. However, this never occurred in our numerical tests.

5.4.7 Case Studies

We consider 2D ceramic shapes made out of beryllium oxide (BeO) under tensile load. Therefore, we set Young's modulus to $\mathtt{E} = 320\,\mathrm{GPa}$ (see, e.g., Munz and Fett, 2001), Poisson's ratio to $\nu = 0.25$, and the ultimate tensile strength to 140 MPa, according to Shackelford et al., 2015. Weibull's modulus is set to $m = 5$, which is on the lower bound of industrial ceramics having m between 5 and 30 as depending on the production process (Morell, 2004). All considered shapes have a fixed length of 1.0 m and a fixed height of 0.2 m on the left and right boundaries. The shapes are fixed on the left boundary, where Dirichlet boundary conditions hold ($\partial\Omega_D$), and on the right boundary, where surface forces may act on and Neumann boundary conditions hold ($\partial\Omega_{N_{\text{fixed}}}$). The upper and lower boundaries are assumed to be force free ($\partial\Omega_{N_{\text{free}}}$). They can be modified within the optimization process. We set $f_{\text{vol}} = 0$ neglecting the gravity forces and $g_{\text{surf}} = 10^7\,\mathrm{Pa}$, representing tensile load. Note that, in order to be consistent with 3D models, we define the force density w.r.t. $\mathrm{Pa} = \mathrm{N/m^2}$ (and not w.r.t. N/m). This is motivated by assuming a constant width of the 2D component of 1 unit (i. e., 1m). Then plane stresses and plane strains are obtained by neglecting Poisson effects in the third dimension.

Here, both starting solution are chosen without involving a decision maker. In the first case, we choose an obviously not efficient solution for a horizontal load transfer to see how the algorithms works. The second test case simulates a shifted load transfer. We thus take an *a posteriori* approach on decision making with regard to design or cost preferences. We are aware that using only one starting design for each test case may bias the solutions of both optimization methods. Nevertheless, numerical experiments with moderately modified initial designs yielded comparable solutions, indicating that in these special cases there is not much to gain by varying the starting designs.

The shapes are discretized by a 41×7 grid (i. e., $n_x = 41$ and $n_y = 7$) using triangles as detailed in Section 5.4.1. The B-spline representation is based on $n_B = 5$ basis functions. Thus, we have in total ten B-spline coefficients. Since the left and right boundary are fixed and we only modify the upper and lower boundary of the components, we have to fix the first and last B-spline coefficients for both, the auxiliary meanline and thickness functions. All in all, we have now six control variables. Moreover, the curvature regularization parameter is set to $\xi = 10^{-4}$, see Section 5.4.5.

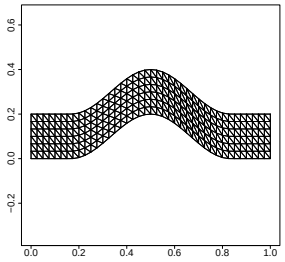
During the optimization process, we monitor the Euclidean norm of the update of the design variables in every iteration and stop when it is lower than 10^{-4}. The implementation is realized in R version 3.5.0 and uses the adjoint finite element code of Bolten, Gottschalk, Hahn, et al., 2019 as a subroutine.

A Straight Joint

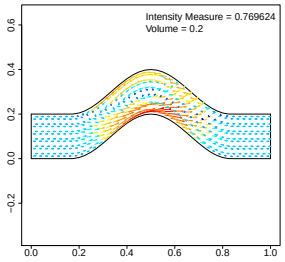
In the first test case, a straight joint is sought that is fixed at the left side, while the tensile load acts on the right side. This is a particularly simple situation where the straight rod connecting from the left to the right can be expected to be optimal, with varying thickness depending on the trade-off between the intensity measure (J_1) and the volume (J_2). The optimization algorithms are challenged by providing a bended beam as a starting shape, which is clearly far from being optimal.

The starting shape is shown in Figure 5.5a, together with the 41×7 tetrahedral discretization X. Its objective values are $J_1(X^{(1)}) = 0.769624$ (intensity measure) and $J_2(X^{(1)}) = 0.2$ (volume), respectively. The relatively high value for the intensity measure J_1 can be explained by the relatively high stresses that are illustrated in Figure 5.5b. Figure 5.5c shows that the B-spline representation based on only five basis functions leads to a rather inaccurate representation, particularly at the left and right boundary. This could be improved by fixing the slopes at the left and right boundary, however, at the price of a significantly reduced design space. Indeed, a majority of the Pareto critical shapes computed during our numerical tests do not have zero slopes at the left and right boundary, particularly in the case of the S-shaped joint considered in the end of Section 5.4.7 below. Note that the smoothing induced by the B-spline representation in this case already leads to dominating objective values of $J_1(\gamma^{(1)}) = 0.453867$ and $J_2(\gamma^{(1)}) = 0.2$.

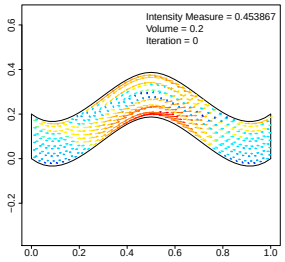
Results Among all shapes with a fixed volume of $J_2(X) = 0.2$, the straight rod shown in Figure 5.5d can be expected to have the minimum possible intensity measure J_1. Indeed, the straight rod shown in Figure 5.5d achieves an objective value of $J_1(X) = 0.00058$. Figures 5.5e and 5.5f show the results of the weighted sum method (Algorithm 3) with weight $\omega = 0.8$ and of the biobjective descent algorithm (Algorithm 4) with scaling parameter $\bar{\omega} = 1.8$. Both methods show a rather quick convergence (with the expected advantage for the biobjective descent algorithm) to solutions that are close to optimal. However, the solution of the biobjective descent algorithm seems to be a local solution with slightly higher stresses (and thus slightly



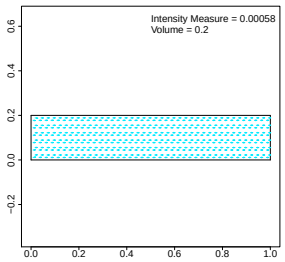
(a) Starting shape: Tetrahedral mesh X



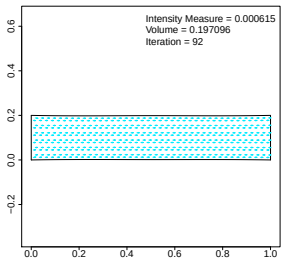
(b) Starting shape: Objective values and stresses



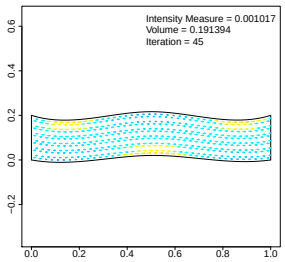
(c) Starting shape: Approximation with B-splines



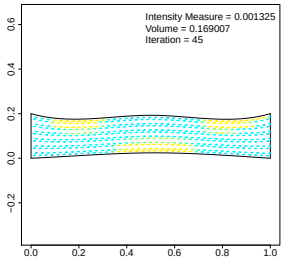
(d) Expected result: Straight rod



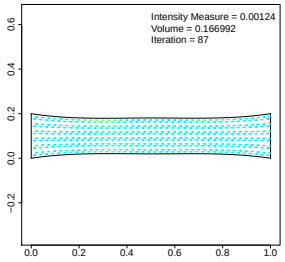
(e) Weighted sum, $\omega = 0.8$



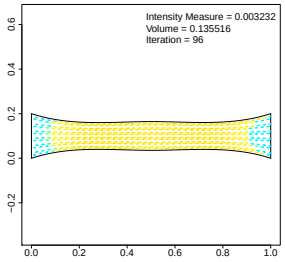
(f) MO descent, $\bar{\omega} = 1.8$



(g) MO descent, $\bar{\omega} = 0.5$



(h) Weighted sum, $\omega = 0.6$



(i) Weighted sum, $\omega = 0.3$

Figure 5.5: Straight joint: Starting solution (row 1), straight rod solutions (row 2), and low volume solutions (row 3).

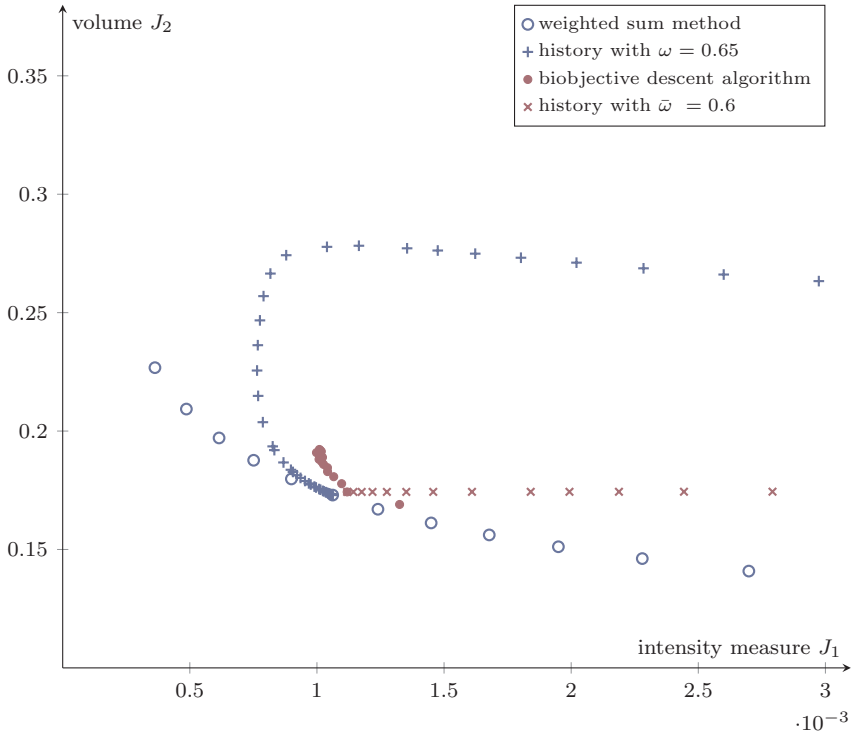


Figure 5.6: Iteration histories of an exemplary run of the weighted sum method (Algorithm 3) and of the biobjective descent algorithm (Algorithm 4). Note that both algorithms use the same starting solution.

higher objective value for J_1).

Figure 5.6 shows iteration histories of exemplary runs of the weighted sum method (Algorithm 3) and of the biobjective descent algorithm (Algorithm 4), respectively. It nicely illustrates that, in contrast to the biobjective descent algorithm, the weighted sum method permits iterations where one objective function deteriorates while the weighted sum objective is still decreasing. This may, in certain situations, help to overcome local Pareto critical solutions. On the other hand, the weighted sum method may get stuck in local minima as well. Indeed, independent of the chosen weight, the histories of the weighted sum method have a similar structure: First mainly the intensity measure (representing the PoF) is improved (since in early stages of the algorithm the gradient of J_1 is considerably larger than the gradient of J_2). Only at later stages of the algorithm, the volume is varied to a larger extent, depending on the given weight.

Note also that the final solution obtained with the biobjective descent algorithm

largely depends on the starting solution, since the objective values can never deteriorate during the optimization process. Thus, when the starting solution has a volume of $J_2(X) = 0.2$, then all Pareto critical shapes that can be computed with the biobjective descent algorithm have a volume of at most 0.2, irrespective of the scaling.

Three shapes with progressively reduced volume (and hence lower cost) are shown in Figures 5.5g to 5.5i. As was to be expected, a lower cost comes at the price of a higher intensity measure (and hence higher PoF). A comparison between Figures 5.5h and 5.5g suggests that also for the low volume solutions, the weighted sum solutions slightly outperform the biobjective descent solutions.

Figure 5.7 summarizes the results of several optimization runs with varying weights (Algorithm 3) and varying scalings (Algorithm 4), respectively. The same starting solution was used in all cases, see Figure 5.5c. While the solution quality of the weighted sum method and of the biobjective descent algorithm is comparable, a clear advantage of the weighted sum method seems to be that it is not so much constrained by the (performance of the) starting solution. Indeed, the weighted sum solutions shown in Figure 5.7 span a large range of alternative objective values in the objective space and thus provide the decision maker with meaningful trade-off information and a variety of solution alternatives.

Both algorithms need in general one gradient computation and k_{step} function evaluations per iteration, where k_{step} is the number of iterations in the Armijo rule to calculate a step length. Additionally, the biobjective descent needs one gradient evaluation, whereas the weighted sum requires one objective function evaluation to determine an initial scaling of the objectives. In this test case the mean number of iterations for the weighted sum method was 94 with around 3.8 Armijo iterations on average. The biobjective descent needed 46 iterations with 1.7 Armijo iterations on average. On this rather coarse grid (41×7) a function evaluation takes about 1.2 seconds and a gradient evaluation around 15.48 seconds, a finer grid would extend the run time significantly. Note that the underlying simulation code for the function evaluation and gradient computation is not optimized w.r.t. runtime. Summing up, an optimization run with the weighted sum method for this test case on a 41×7 grid took about 31.4 minutes on average. The biobjective descent algorithm took about 14.9 minutes on average. All algorithms are implemented in R version 3.5.0, and the numerical tests run on a PC with Intel Core i7-8700 CPU @ 3.20 GHz, 31.2 GB RAM.

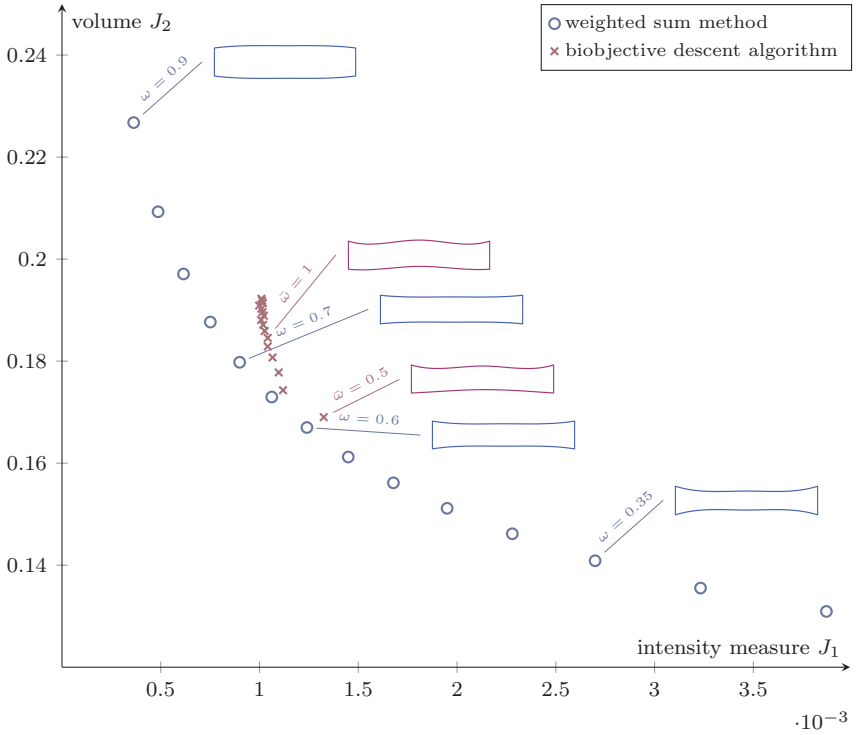


Figure 5.7: Approximated non-dominated front for the straight joint. The associated Pareto critical shapes are shown for selected weightings/scalings.

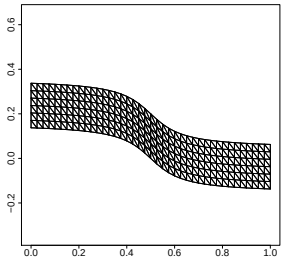
An S-Shaped Joint

A more complex situation is obtained when the left and right boundaries are not fixed at the same height, i. e., when an S-shaped joint is to be designed. In our tests, we fix the right boundary about 0.27 m lower than the left boundary. The starting shape and its 41×7 tetrahedral discretization X, that is used for all optimization runs, is shown in Figure 5.8a. Figure 5.8b highlights the stresses that are particularly strong towards the left boundary. The respective objective values are $J_1(X^{(1)}) = 1.520058$ (intensity measure) and $J_2(X^{(1)}) = 0.2$ (volume), respectively. As can be expected, the intensity measure (and hence also the PoF) is considerably higher than in the case of the straight joint discussed above. Despite the significant smoothing induced by the B-spline representation of the initial shape shown in Figure 5.8c, it has an even higher value of the intensity measure of $J_1(\gamma^{(1)}) = 1.910532$ (and hence a higher PoF value), while $J_2(\gamma^{(1)}) = 0.2$ remains constant.

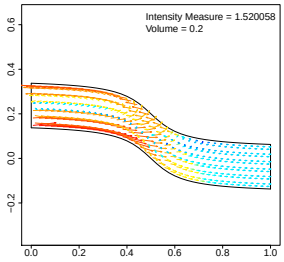
Results We observe that the resulting shapes resemble the profile of a whale. If we consider 1st principal stress of the stress tensor on the grid points of the initial shape resulting from tensile load, see Figure 5.8c, we observe an anti clockwise eddy in the left part of the joint. The hunch close to the left boundary of the optimized shapes gives room for the occurring stresses and therefore improves the intensity measure and, likewise, the PoF.

Note that different from the case of the straight rod, we have no prior knowledge on the Pareto optimal shapes. For the solutions shown in Figures 5.8d and 5.8e, we can only guarantee that they are (approximately) Pareto critical, i. e., the respective optimization runs terminated due to the criticality test. Figure 5.8f shows a shape with a significantly higher volume of $J_2(X) = 0.225906$, and with a largely improved intensity measure of $J_1(X) = 0.196791$. This shape was obtained with the weighted sum method with weight $\omega = 0.85$ after 150 iterations. In this case, the algorithm terminated since it reached the maximum number of iterations and not due to convergence. We observed that all optimization runs of the weighted sum method with $\omega \geq 0.85$ were not converging in this setting. Thus in these cases it is not guaranteed, that the resulting solutions are Pareto critical. Note that, given a starting solution with a volume of 0.2, this shape is not attainable with the biobjective descent algorithm.

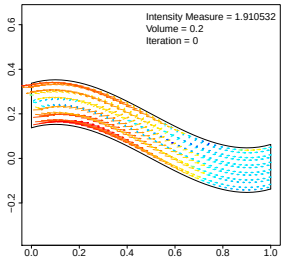
However, there is no guarantee that the computed shapes are Pareto optimal. For example, the shape shown in Figure 5.8e obtained with the weighted sum method with weight $\omega = 0.8$ achieves objective values of $J_1(X) = 0.293853$ and $J_2(X) = 0.188445$,



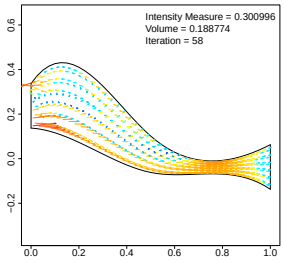
(a) Starting shape: Tetrahedral mesh X



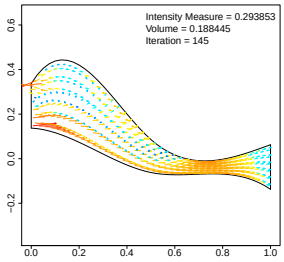
(b) Starting shape: Objective values and stresses



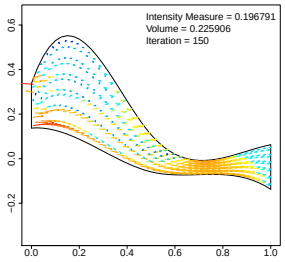
(c) Starting shape: Approximation with B-splines



(d) MO descent, $\bar{\omega} = 1.1$



(e) Weighted sum, $\omega = 0.8$



(f) Weighted sum, $\omega = 0.85$, not converged

Figure 5.8: S-shaped joint: Starting solution (row 1), two exemplary Pareto critical solutions (5.8d and 5.8e) and a not converged solution of the weighted sum method (5.8f).

and hence slightly dominates the shape shown in Figure 5.8d obtained with the biobjective descent algorithm with scaling parameter $\bar{\omega} = 1.1$ that has objective values $J_1(X) = 0.300996$ and $J_2(X) = 0.188774$.

Figure 5.9 summarizes the results of several optimization runs of both Algorithms 3 and 4 in the objective space. Note that not all solutions of the weighted sum method lie on the convex hull of the computed points (and are thus not globally optimal for a weighted sum scalarization). In some cases, the biobjective descent algorithm also computes dominated points, while in other cases it found solutions that lie even below the convex hull of the weighted sum solutions (see, e.g., the result for $\bar{\omega} = 0.5$ in Figure 5.9).

A larger range of alternative objective vectors is, as in the case of the straight rod, obtained with the weighted sum method. A cross-test between the two methods, where the final solution of Algorithm 3 was used as starting solution for Algorithm 4, confirms that local Pareto critical solutions were found for $\omega \leq 0.8$.

Compared to test case 1, the optimization runs for test case 2 needed in general more iterations. The mean number of iterations for the weighted sum method in test case 2 was 107 with around 5.3 Armijo iterations on average. The biobjective descent needed 74 iterations with 3.9 Armijo iterations on average. Thus, the weighted sum method needed about 39.06 minutes on average and the biobjective descent algorithm took about 26.96 minutes on average.

In the meantime, Bolten, Doganay, et al., 2024 have shown that better shapes can be calculated for this test case using momentum methods.

5.5 Implementation with Structured Meshes

The second implementation of scalarization techniques for the biobjective shape optimization problem (5.8) is based on another type of finite element discretization. Here we use structured finite element grids for evaluation of the state equation (5.2) and the objective functions given in (5.6) and (5.7). This implementation allows us a parameter-free representation of the shapes. We implemented the weighted sum method and the hypervolume method based on this discretization approach in the programming language Python 3.

We start in Section 5.5.1 with a description of the finite element discretization which was implemented by Camilla Hahn, see Hahn, 2021. Afterwards we consider the hypervolume scalarization (HV-SP) of the biobjective shape optimization problem

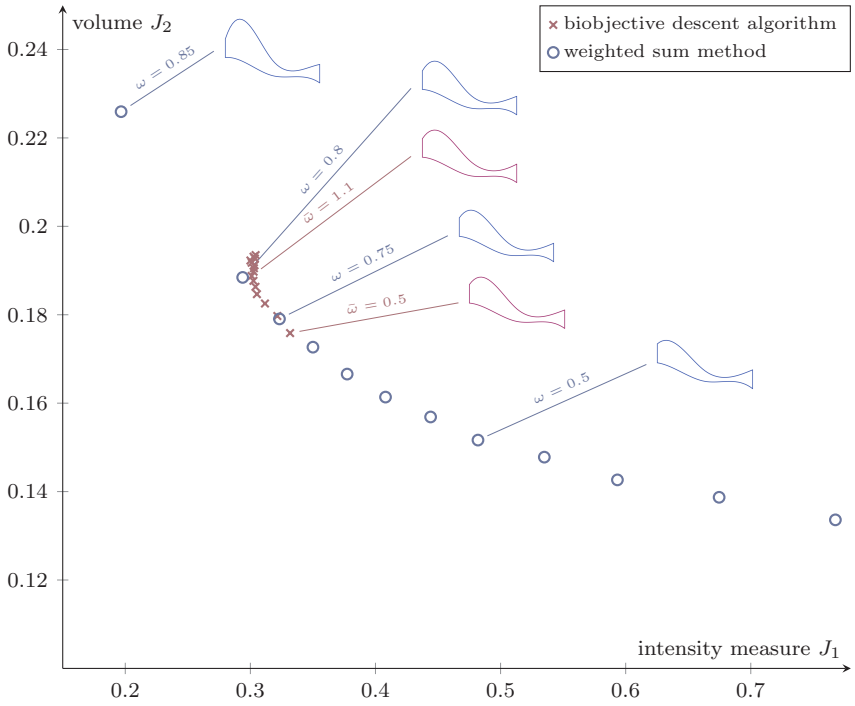


Figure 5.9: Outcome vectors for the S-shaped joint. The associated Pareto critical shapes are shown for selected weightings/scalings.

(5.8) which we solve using an iterative ascent algorithm in the framework of a *'first discretize, then optimize'* approach. Details of the numerical implementation are presented including the discretization approach the determination of ascent directions for the hypervolume scalarization and the update scheme for the current shape. Furthermore, the choice of the reference point is discussed. The parts mentioned above are already published in Schultes et al., 2021.

5.5.1 Finite Element Discretization

The component to be optimized (i. e., the current shape) is discretized via finite elements on a structured grid in the following way. Consider a rectangular domain $\hat{\Omega}$ with width $\hat{\Omega}_l$ and height $\hat{\Omega}_h$ that entirely covers the initial shape $\Omega^{(0)}$ (see Figure 5.1 for an illustration). The domain is discretized by a regular triangular grid denoted by $\hat{T}$ with n_x and n_y grid points in x- and y-direction, respectively. The corresponding mesh sizes are given by $m_x = \hat{\Omega}_l/(n_x - 1)$ and $m_y = \hat{\Omega}_h/(n_y - 1)$. The mesh sizes should be chosen such that $m_x \approx m_y$ in order to avoid distorted finite elements. In a next step, as visualized in Figure 5.10, the boundary $\partial\Omega^{(0)}$ of the initial shape is superimposed onto the grid and the closest grid points are moved onto the boundary in a controlled way, such that degenerate elements do not occur. More precisely, it is guaranteed that the maximum angle condition (Babuška and Aziz, 1976) is satisfied. The adapted grid is denoted by T_0^∞. The computations are performed only on that part of the grid that represents $\Omega^{(0)}$ which is called the *active grid* and denoted by T_0, see Figure 5.10.

In the iterative optimization process (see Section 5.5.2 for more details) the active grid T_{k+1} of the next iteration, $k = 0, 1, 2, \ldots$, representing the updated shape $\Omega^{(k+1)}$, is generated by starting again from $\hat{T}$. Consequently, the connectivity is maintained throughout all iterations. Rather than going through all elements (as required in the first iteration) we now incorporate the information from the previous mesh T_k^∞. In the 'updating procedure' it is only necessary to iterate over elements in a certain neighborhood of previous boundary elements, and adapt them if necessary. Note that a complete re-meshing is necessary only when the next iterate $\Omega^{(k+1)}$ differs too much from $\Omega^{(k)}$, which is usually only the case when the update step exceeds the mesh size in at least one grid point. This property and the persistent connectivity lead to a significant speed up compared to standard re-meshing techniques. As stated above, in most iterations the majority of the elements inside the component remain unchanged. This leads to an additional advantage over both standard re-meshing and

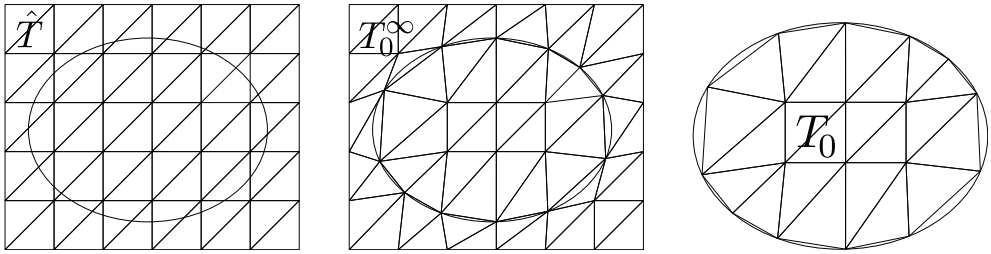


Figure 5.10: Exemplary adaption of the finite element grid to a shape $\Omega^{(0)}$ in iteration $k = 0$.

mesh morphing approaches. In regions where elements are not changed, the entries of the governing PDE are also not changed. Therefore, a full re-assembly of the system matrix and the right hand side is avoided and only entries corresponding to modified elements have to be updated.

In the following, we distinguish between the continuous shape $\Omega^{(k)}$ and its (approximative) representation on the finite element grid. The discretized shape, induced by the active grid T_k, is denoted by $\Omega^{(k)}_{n_x \times n_y}$. This implicates that the gradient of the hypervolume indicator ∇H is evaluated w.r.t. the discretized shape, i. e., $\nabla H(\Omega^{(k)}_{n_x \times n_y})$ is defined on the grid points given by T_k.

5.5.2 Iterative Ascent Algorithm for the Hypervolume Scalarization

The hypervolume scalarization problem of the biobejctive optimization problem (5.8) is given by

$$\begin{aligned} \max \quad & (r_1 - J_1(\Omega)) \cdot (r_2 - J_2(\Omega)) \\ \text{s. t.} \quad & J_1(\Omega) \leqq r_1, \\ & J_2(\Omega) \leqq r_2, \\ & \Omega \in \mathcal{O}^{\text{ad}} \end{aligned} \tag{5.18}$$

with reference point $r = (r_1, r_2)^\top \in \mathbb{R}$.

Ascent direction and update scheme In the following we develop an iterative update scheme for the hypervolume scalarization of the biobjective shape optimization problem (5.8). As stated above, the finite element discretization of the component

is based on a regular grid, where only the grid points that are close to the boundary of the current shape $\Omega^{(k)}$ are adapted to the boundary of the component. All finite elements that are not adjacent to the boundary of the current shape $\Omega^{(k)}$ remain unchanged and form a subset of the regular grid. Whenever the component is updated, we thus want to modify the boundary points of the component without moving inner or outer grid points. For this purpose we consider the grid points lying on the boundary of the shape $\Omega^{(k)}$, or rather of its finite element representation $\Omega^{(k)}_{n_x \times n_y}$. Since these grid points are induced by the active grid T_k, we call them *boundary points* and denote them by ∂T_k.

The discretized shape $\Omega^{(k)}_{n_x \times n_y}$ in iteration k is updated by determining a search direction and calculating a step length. For a given search direction $d^{(k)}$ and step length t_k, we define the update of the component by the expression

$$\Omega^{(k+1)}_{n_x \times n_y} = \Omega^{(k)}_{n_x \times n_y} \oplus t_k \cdot d^{(k)}. \tag{5.19}$$

The operation $\oplus$ is defined as follows: First, the boundary points are moved by $t_k \cdot d^{(k)}$ resulting in

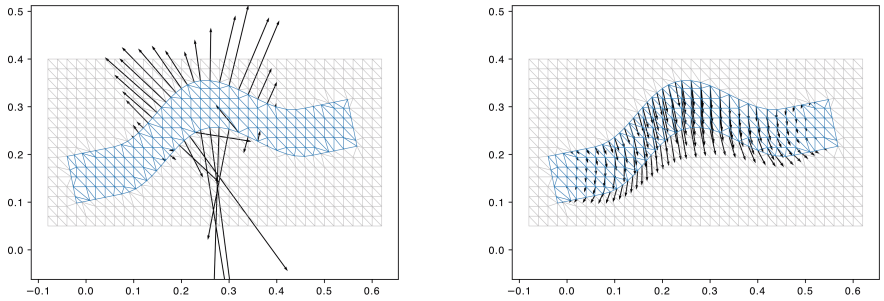
$$\partial T'_k \coloneqq \partial T_k + t_k \cdot d^{(k)}.$$

Then $\partial T'_k$ is fitted by cubic splines, resulting in the updated shape $\Omega^{(k+1)}$. Finally, the new active grid T_{k+1} is generated based on $\partial\Omega^{(k+1)}$ as detailed in Section 5.5.1. This defines $\Omega^{(k+1)}_{n_x \times n_y}$ in (5.19). Note that the boundary points ∂T_{k+1} of iteration $(k+1)$ do not have to coincide with $\partial T'_k$ since T_{k+1} is generated based on $\partial\Omega^{(k+1)}$, which is defined by cubic splines.

The search direction $d^{(k)}$ is computed based on the gradient of the hypervolume indicator function $\nabla H(\Omega^{(k)}_{n_x \times n_y})$. However, due to the discretization $\nabla H(\Omega^{(k)}_{n_x \times n_y})$ is usually largely irregular and needs to be smoothed to avoid zigzagging boundaries and overfitted spline representations of the next iterate. To obtain a smooth deformation field for the boundary points, we follow the approach suggested in Schmidt and Schulz, 2009 and use a *Dirichlet-to-Neumann* map to compute a smoothed gradient. Let $\tilde{\nabla} H(\Omega^{(k)}_{n_x \times n_y})$ denote the smoothed gradient. Then

$$d^{(k)} \coloneqq \tilde{\nabla} H(\Omega^{(k)}_{n_x \times n_y})\Big|_{\partial T_k} \tag{5.20}$$

denotes its values on the boundary points. See Figure 5.11 for a visual comparison of



(a) $\nabla H(\Omega^{(k)}_{n_x \times n_y})$. (b) $\tilde{\nabla} H(\Omega^{(k)}_{n_x \times n_y})$.

Figure 5.11: Unsmoothed (left) and smoothed (right) gradient of the hypervolume indicator.

the gradient $\nabla H(\Omega^{(k)}_{n_x \times n_y})$ and the smoothed gradient $\tilde{\nabla} H(\Omega^{(k)}_{n_x \times n_y})$ of an exemplary shape.

The step length t_k in direction $d^{(k)}$ is determined according on the Armijo-rule (see, e.g., Bazaraa, 2006; Geiger and Kanzow, 1999). For given parameters $\sigma, \beta \in (0,1)$ we set

$$t_k := \max_{\ell \in \mathbb{N}_0} \left\{ \beta^\ell : H\big(\Omega^{(k)}_{n_x \times n_y} \oplus \beta^\ell \, d^{(k)}\big) \geq H(\Omega^{(k)}_{n_x \times n_y}) + \sigma \, \beta^\ell \, \nabla H(\Omega^{(k)}_{n_x \times n_y})^\top d^{(k)} \right\}. \tag{5.21}$$

The complete iterative ascent algorithm using smoothed gradients is summarized in Algorithm 5. As stopping conditions, an upper bound K on the number of iterations, an upper bound on the number of iterations ℓ in the evaluation of the Armijo rule (5.21), and a lower bound ε on the Frobenius norm of the search direction are used.

Re-scaling the search direction. In order to avoid re-meshing operations whenever possible, it is often advantageous to re-scale the search direction such that the gird T^∞_{k+1} can be computed by performing a simple grid adaption (c.f. Section 5.5.1). Indeed, when for example for a large gradient (e.g., resulting from large given loads) the step $t_k = 1$ is not feasible, this leads to numerical problems when implementing Algorithm 5. To avoid such situations, the search direction $d^{(k)}$ is re-scaled in the following way: First, the norm $\|d^{(k)}_{ij}\|_2$ is calculated for all grid points (i,j), $i = 1, \ldots, n_x$ and $j = 1, \ldots, n_y$, where $d^{(k)}_{ij}$ is the entry of $d^{(k)}$ for grid point (i,j). Since $d^{(k)}$ is

Data: Starting solution $\Omega^{(0)}_{n_x \times n_y}$, reference point r; Armijo parameters $\sigma, \beta \in (0,1)$, accuracy $\varepsilon > 0$, maximal number of iterations $K \in \mathbb{N}$;
Result: Approximate solution of the hypervolume problem $\Omega^{*}_{n_x \times n_y}$;
Set $k := 0$, calculate $\tilde{\nabla} H(\Omega^{(0)}_{n_x \times n_y})|_{\partial T_k}$;
while $\|\tilde{\nabla} H(\Omega^{(k)}_{n_x \times n_y})|_{\partial T_k}\| < \varepsilon$ *and* $k \leq K$ **do**
  Compute a search direction $d^{(k)} := \tilde{\nabla} H(\Omega^{(k)}_{n_x \times n_y})\big|_{\partial T_k}$;
  Compute a step length $t_k \in (0,1]$ according to the Armijo-rule (5.21);
  $\Omega^{(k+1)}_{n_x \times n_y} := \Omega^{(k)}_{n_x \times n_y} \oplus t_k \cdot d^{(k)}$;
  $k := k + 1$;
end
$\Omega^{*}_{n_x \times n_y} := \Omega^{(k)}_{n_x \times n_y}$;

Algorithm 5: Iterative optimization scheme for hypervolume maximization.

only defined at the boundary points, $d^{(k)}_{ij}$ is set to zero for grid points (i,j) that are not related to the boundary points. When $\max_{i,j} \|d_{ij}\|_2$ is larger than a predefined percentage $p_{\max}$ of the minimum mesh size $\min\{m_x, m_y\}$, with $p_{\max} < 1$, then the search direction is re-scaled by the factor $s(p_{\max})$ given by

$$s(p) = \frac{p}{\max_{i,j} \|d_{ij}\|_2} \min\{m_x, m_y\}. \tag{5.22}$$

By computing a step length $t_k \in (0,1]$ it is then guaranteed that the boundary points do not move more than the mesh size, and a grid apdation can be performed.

Furthermore, numerical tests have shown that near an efficient solution the search direction $d^{(k)}$ becomes very small, thus slowing down the potential improvement while the unsmoothed gradient $\nabla H(\Omega^{(k)}_{n_x \times n_y})$ may still be large. Then the search direction is re-scaled when $\max_{i,j} \|d_{ij}\|_2$ is less than a given percentage $p_{\min} \in (0,1)$ of the mesh size. In this case, the search direction is multiplied by $s(p_{\min})$.

Grid refinement In order to accelerate the convergence to the Pareto front, it may be useful to start the optimization on a coarse grid and later continue on a finer grid. Since for numerical reasons each update step is restricted to the mesh size, we expect that a coarse grid allows for larger steps in early stages of the optimization procedure. Moreover, the evaluation of objective function values and gradients on the coarse grid is generally faster. On the other hand, with a finer resolution of the grid the accuracy of the approximation of objective function values and gradients is improved. Thus,

near an efficient solution large steps are not necessary and a finer grid may result in a better approximation of the shape.

5.5.3 Choice and Variation of the Reference Point

For the iterative ascent algorithm (Algorithm 5) a feasible starting solution $\Omega^{(0)}_{n_x \times n_y}$ and a reference point $r \in \mathbb{R}^2$ with $r > J(\Omega^{(0)}_{n_x \times n_y})$ are needed as input. Since in general neither the Pareto front nor efficient shapes are known beforehand, the reference point can only be chosen based on a known feasible shape $\Omega^{(0)}_{n_x \times n_y}$ that may actually be far from the Pareto front. To ensure that the hypervolume scalarization (5.18) is well defined and that the constraints $J_1(\Omega) \leq r_1$ and $J_2(\Omega) \leq r_2$ are redundant throughout the execution of Algorithm 5 (c.f. Chapter 3 for a discussion of this issue,) we select the reference point r such that

$$r = \left(J_1(\Omega^{(0)}_{n_x \times n_y}) + \delta_1,\ J_2(\Omega^{(0)}_{n_x \times n_y}) + \delta_2 \right)^\top \tag{5.23}$$

with positive parameters $\delta_1,\ \delta_2 > 0$.

The impact of the reference point on the hypervolume indicator was already discussed in Auger et al., 2009. They derive properties of *optimal μ distributions* depending on the choice of the reference point, i. e., of representative subsets of the Pareto front of cardinality μ maximizing the joint hypervolume. Moreover, they derive an explicit lower bound for the reference point guaranteeing that, under appropriate conditions, the extreme solutions are contained in the representation.

In order to take advantage of the fact that, at least theoretically, the complete Pareto front can be generated by varying the reference point in the hypervolume scalarization (5.18) (see Corollary 7), we suggest a systematic approach to approximate the Pareto front of problem (5.8) by repeatedly solving problem (5.18) with *different* reference points. The ultimate goal is the determination of a *Pareto front generating* reference set (PFG reference set for short). Let $\mathcal{Y}_{\mathrm{H}}(r)$ denote the optimal outcome vector obtained from solving problem (5.18), and let $\mathcal{Y}_{\mathrm{H}}(R) \coloneqq \bigcup_{r \in R} \mathcal{Y}_{\mathrm{H}}(r)$ denote the set of all optimal outcome vectors obtained over all reference points in the set $R \subseteq \mathbb{R}^2$, called *reference set*.

Definition 11. *A set $R \subseteq \mathbb{R}^2$ is called* Pareto front generating (PFG) *reference set if and only if $\mathcal{Y}_N = \mathcal{Y}_{\mathrm{H}}(R)$. Moreover, we denote a reference set $R \subseteq \mathbb{R}^2$ as* pPFG reference set*, if and only if it is generating the set of properly non-dominated outcome vectors, i. e., $\mathcal{Y}_{\mathrm{PND}} = \mathcal{Y}_{\mathrm{H}}(R)$.*

Note that a PFG reference set always exists since, for example, $R = \mathcal{Y}_{\mathrm{N}}$ is a valid selection, c.f. Corollary 7. Note also that a PFG reference set may contain redundant reference points that can be omitted without loosing the PFG property.

From a practical point of view, we can not expect to solve the hypervolume scalarization (5.18) to global optimality and thus only aim at an approximation of the Pareto front $\mathcal{Y}_{\mathrm{N}}$. Towards this end, we suggest to choose the reference set R based on varying values of $\delta = (\delta_1, \delta_2) > (0, 0)$ in (5.23). More precisely, we suggest to set $(\delta_1, \delta_2) = (\xi, \frac{1}{\xi})$ with $\xi > 0$ in (5.23), i. e.,

$$r(\xi) \coloneqq \left(J_1(\Omega^{(0)}_{n_x \times n_y}) + \xi,\ J_2(\Omega^{(0)}_{n_x \times n_y}) + \frac{1}{\xi} \right)^\top \tag{5.24}$$

and set $R_{\mathrm{H}} \coloneqq \{r(\xi) \in \mathbb{R}^2 \colon \xi > 0\}$. This ensures that the starting solution $\Omega^{(0)}_{n_x \times n_y}$ is strictly feasible for the hypervolume scalarization (5.18) for all choices of $r(\xi) \in R_{\mathrm{H}}$, i. e., $J(\Omega^{(0)}_{n_x \times n_y}) < r(\xi)$ for all $\xi > 0$. This implies that $R_{\mathrm{H}} \subset J(\Omega^{(0)}_{n_x \times n_y}) + \mathbb{R}^2_>$ with $\mathbb{R}^2_> = \{y \in \mathbb{R}^2 \colon y_i > 0,\ i = 1, 2\}$. Hence, $\mathcal{Y}_{\mathrm{N}} \cap R_{\mathrm{H}} = \emptyset$, i. e., the Pareto front lies completely 'below' the reference set.

For reference points from the reference set R_{H} the result of Theorem 6 can be strengthened for the biojective case.

Lemma 12. *An optimal solution of the hypervolume scalarization* (5.18) *w.r.t. a reference point from the set R_{H} is a properly efficient solution of the corresponding biobjective optimization problem* (5.8)*.*

Proof. Consider an optimal solution of the hypervolume scalarization (5.18) $y = J(\bar{\Omega}) \in \mathcal{Y}_{\mathrm{H}}(\bar{r})$ for an arbitrary reference point $\bar{r} \in R_{\mathrm{H}}$. Since $J(\Omega^{(0)}_{n_x \times n_y}) < \bar{r}$ we know that for this reference point $H(\Omega^{(0)}_{n_x \times n_y}) > 0$, and hence also $H(\bar{\Omega}) > 0$. Therefore, the hypervolume rectangle induced by y and $\bar{r}$ must have positive side lengths $a(\bar{\Omega}) > 0$ and $b(\bar{\Omega}) > 0$. This implies that y is locally optimal for a weighted sum scalarization with weights $\lambda_1 = b(\bar{\Omega}) > 0$ and $\lambda_2 = a(\bar{\Omega}) > 0$ since this has the same cone of improving directions as the hypervolume indicator in this point (see Section 3.4.1 for a detailed derivation). It follows that $y \in \mathcal{Y}_{\mathrm{PND}}$. □

Note that Lemma 12 is in line with the results of Auger et al., 2009 who showed that, irrespective of the choice of the reference point, non-dominated points with unbounded trade-off are not part of what they call 'optimal μ distributions', that is, of subsets of points on the Pareto front that maximize the joint hypervolume.

Note also that by varying $\xi > 0$ different reference points are obtained that lie along

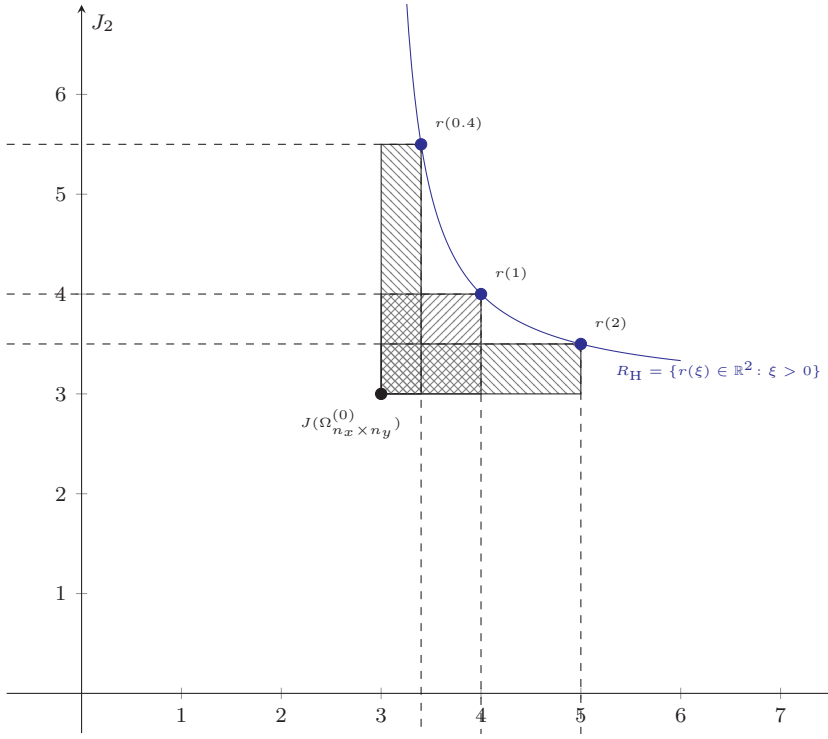


Figure 5.12: Reference set R_{H} defining a hyperbola in the objective space.

a hyperbola in the objective space, see Figure 5.12 for an illustration. Indeed, for two parameter values $0 < \xi_1 < \xi_2$ we have that $r_1(\xi_1) < r_1(\xi_2)$ and $r_2(\xi_1) > r_2(\xi_2)$. Hence, the hypervolume rectangle spanned by $J(\Omega^{(0)}_{n_x \times n_y})$ and $r(\xi_1)$ cannot be contained in the hypervolume rectangle spanned by $J(\Omega^{(0)}_{n_x \times n_y})$ and $r(\xi_2)$ and vice versa. However, for all associated hypervolume rectangles it holds that $a(\Omega^{(0)}_{n_x \times n_y}) \cdot b(\Omega^{(0)}_{n_x \times n_y}) = 1$.

While we can not guarantee that the set R_{H} is a PFG reference set in general, we argue that the chances are high that the reference set R_{H} induces solutions with varying objective values. This is validated by the numerical experiments described in Section 5.5.4 below. From a theoretical perspective, we argue that at least for convex problems the reference set R_{H} is pPFG, i. e., the properly efficient set can be generated by varying the reference points in R_{H}.

Theorem 13. *Suppose that for $p = 2$ the biobjective optimization problem* (MOP) *is convex, i. e., all efficient solutions of* (MOP) *are supported efficient solutions. Then $\mathcal{Y}_{\mathrm{PND}} = \mathcal{Y}_{\mathrm{H}}(R_{\mathrm{H}})$, i. e., R_{H} is a pPFG reference set.*

Proof. First, recall that Lemma 12 implies that $\mathcal{Y}_{\mathrm{H}}(R_{\mathrm{H}}) \subseteq \mathcal{Y}_{\mathrm{PND}}$. Thus, it remains to be shown that $\mathcal{Y}_{\mathrm{PND}} \subseteq \mathcal{Y}_{\mathrm{H}}(R_{\mathrm{H}})$.

Now let $y = J(\bar{\Omega}) \in \mathcal{Y}_{\mathrm{PND}}$ be properly efficient. Then there exist weights $\lambda_1, \lambda_2 > 0$ such that $\bar{\Omega}$ is optimal for the associated weighted sum scalarization (5.14), see, e.g., Ehrgott, 2005; Geoffrion, 1968. Now define a rectangle with sides $a \coloneqq \lambda_2 > 0$ and $b \coloneqq \lambda_1 > 0$ and let $r' \coloneqq J(\bar{\Omega}) + (a, b)^\top$. It is easy to see that the hypervolume rectangle spanned by $J(\bar{\Omega})$ and r' is precisely the rectangle with side lengths a and b. From the discussion in Section 3.4.1 we can conclude that $\bar{\Omega}$ is optimal for the hypervolume scalarization (HV-SP) with reference point r' since in this situation the set of improving points w.r.t. the hypervolume indicator is completely contained in that of the weighted sum (which is empty in this case). See Figure 5.13 for an illustration of the respective contour lines. Moreover, $J(\bar{\Omega})$ is the unique optimal outcome vector of the hypervolume scalarization (HV-SP) in this case. Since this property only relies on the ratio $\frac{b}{a}$ of the side lengths of the hypervolume rectangle, the reference point can be moved along the half-line starting at $J(\bar{\Omega})$ and passing through r' until it intersects the set R_{H}. This intersection point exists and can be determined as the positive solution ξ of the system

$$\begin{aligned} J_1(\bar{\Omega}) + \tau\, a &= J_1(\Omega^{(0)}_{n_x \times n_y}) + \xi \\ J_2(\bar{\Omega}) + \tau\, b &= J_2(\Omega^{(0)}_{n_x \times n_y}) + \frac{1}{\xi} \end{aligned}$$

which yields

$$\xi^2 - \xi \underbrace{\left((J_1(\bar{\Omega}) - J_1(\Omega^{(0)}_{n_x \times n_y})) - \frac{a}{b}(J_2(\bar{\Omega}) - J_2(\Omega^{(0)}_{n_x \times n_y})) \right)}_{=:c} - \frac{a}{b} = 0.$$

We obtain two solutions for ξ, namely

$$\xi_{1,2} = \frac{c}{2} \pm \sqrt{\frac{c^2}{4} + \frac{a}{b}}.$$

Since $\frac{a}{b} > 0$ we have that $(\frac{c^2}{4} + \frac{a}{b})^{\frac{1}{2}} > |\frac{c}{2}|$ and hence exactly one of the two solutions for ξ is positive, irrespective of the sign of c. This solution yields the sought reference point $r(\xi) \in R_{\mathrm{H}}$, and we can conclude that $\mathcal{Y}_{\mathrm{PND}} \subseteq \mathcal{Y}_{\mathrm{H}}(R_{\mathrm{H}})$. □

Note that even though Theorem 13 is restricted to convex problems, the set R_{H} may be pPFG also for non-convex problems. This depends on the degree of non-convexity on one hand, and on the distance of the reference set R_{H} from the Pareto front in the

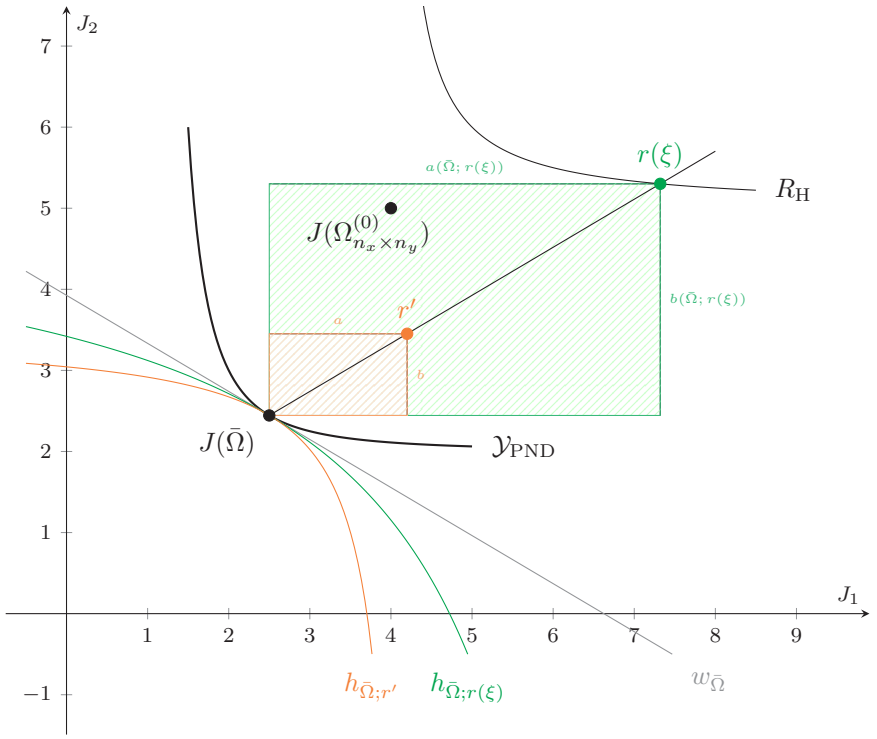


Figure 5.13: Identification of a reference point $r(\xi) \in R_{\mathrm{H}}$ that generates a given outcome vector $J(\bar{\Omega})$.

objective space on the other hand, where the latter is defined by the image of the starting solution $J(\Omega^{(0)}_{n_x \times n_y})$.

5.5.4 Hypervolume solutions with structured meshes

We consider a 2D test case of a ceramic rod consisting of Beryllium oxide (BeO). The rod is 0.6 m long and has a height of 0.1 m. It is fixed on one side while tensile load is applied on the other side, modeling a horizontal load transfer. On the fixed boundary Ω_D of the rod the displacement field $u(x)$ is zero, i. e., here Dirichlet boundary conditions hold, c.f. (5.2). The part $\Omega_{N_{\text{fixed}}}$ denotes the boundary on which the surface loads g are applied, i. e., where Neumann boundary conditions hold. The remaining parts of the boundary, denoted by $\Omega_{N_{\text{free}}}$, can be modified according to the optimization objectives whereas the boundaries Ω_D and $\Omega_{N_{\text{fixed}}}$ are fixed during the optimization. In particular, the corner points cannot be moved. In Figure 5.1 the rod and the decomposition of its boundary into Ω_D, $\Omega_{N_{\text{fixed}}}$ and $\Omega_{N_{\text{free}}}$ are illustrated.

As we consider the same ceramic material BeO as in Doganay et al., 2020, see Section 5.4, we use the same material parameter setting in order to evaluate the state equation (5.2) and the PoF objective (5.6) In particular, Young's modulus is set to E = 320 GPa, Poisson's ratio to $\nu = 0.25$, the ultimate tensile strength to 140 MPa and the Weibull parameter to $m = 5$. The surface loads, acting on $\Omega_{N_{\text{fixed}}}$, are set to $g = (10^8, 0)^\top$ Pa and the volume force density is set to $f = (0, 1000)^\top$ Pa. To speed up the numerical evaluation of the state equation, an improved discretization approach based on regular grids is employed, see Section 5.5.1 for more details and Figure 5.14 for an illustration of the considered shape. In contrast to Section 5.4, volume forces that model, e.g., gravity, are included in the numerical evaluation.

We choose a bent rod as shown in Figure 5.1 as the initial shape (i. e., as the starting solution) for the optimization that is implemented using Algorithm 5. Note that this shape is clearly not efficient. Indeed, in this situation the efficient shapes are straight rods of different width, trading off between small probabilities of failure and high volumes on one hand, and high probabilities of failure and low volumes on the other hand. This simple setting has the advantage that the optimization results can be validated and compared with the 'true' Pareto front.

The rectangular domain $\hat{\Omega}$ that contains all admissible shapes $\mathcal{O}^{\text{ad}}$ is given by a rectangle of width $\hat{\Omega}_l = 0.7$ m and height $\hat{\Omega}_h = 0.35$ m. The initial shape is placed in the interior of this rectangle. Moreover, the whole setting is slightly rotated so that the boundaries of the initial shape can be represented by interpolating cubic

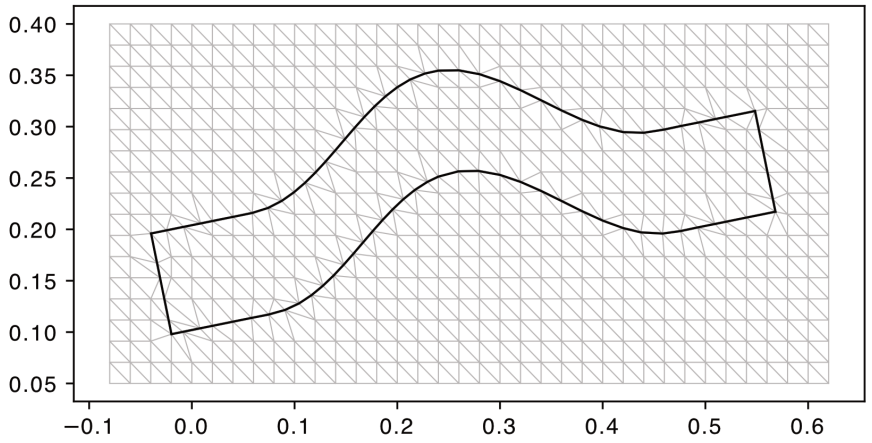


Figure 5.14: Illustration of the discretized rod with 36×18 grid points.

splines. The domain $\hat{\Omega}$ is discretized by a triangular grid with $n_x \times n_y$ grid points as described in Section 5.5.1. Hereby, we determine the number of grid points such that a prescribed mesh size of $m_x = m_y \approx 0.02\,\mathrm{m}$ is realized. In this setting, this results in a discretization with 36×18 grid points, see Figure 5.14. Note that this shape representation has more degrees of freedom than the B-spline representation in Section 5.4. This significantly increases the flexibility of this approach.

In the numerical tests reported below, we use the Armijo-rule (5.21) with parameters $\beta = 0.5$ and $\sigma = 0.1$. Moreover, the smoothed gradient $d^{(k)}$ is adaptively scaled such that $\max_{i,j} \|d_{ij}^{(k)}\|_2$ is within 10% to 90% of the mesh size, i. e., $p_{\min} = 0.1$ and $p_{\max} = 0.9$ as explained in Section 5.5.2. The optimization process is restricted to 500 iterations, i. e., $K = 500$. To avoid tiny step sizes, the Armijo-rule is restricted to 30 iterations, i. e., $\ell \in \{0, \dots, 30\}$. Whenever the Armijo rule does not yield a sufficient improvement for reasonable step sizes, the algorithm stops. Moreover, the algorithm stops when the Frobenius norm of the unscaled search direction $d^{(k)}$ is lower than $\varepsilon = 10^{-10}$.

Twelve exemplary reference points $r(\xi)$ are selected from the pPFG reference set R_{H} (see (5.24)), with $\xi \in \{0.0005, 0.001, 0.0015, \dots, 0.005, 0.006, 0.007\}$. Algorithm 5 is implemented in Python 3.6 and all tests are run under opensuse 15.0 on an IntelCore i7-8700 with 32 GB RAM.

Results

The twelve optimization runs with varying reference points yield 12 different mutually non-dominated outcome vectors. The resulting approximation of the Pareto front is shown in Figure 5.15. As was to be expected, the optimized shapes resemble a straight rod with varying thickness. The objective vector of the initial shape $\Omega_{36\times18}^{(0)}$ is $(18456.28, 0.06)^\top$, which is actually far from the Pareto front.

Several exemplary shapes are shown in Figure 5.15. In addition to the illustration of the respective shapes, the stresses in the direction of the tensile load acting on the finite elements are depicted as arrows at the respective center points. Note that the diagonal pattern of the arrows is due to the rotated (not axis-parallel) grid structure.

It turns out that the iterates evolve similarly for different optimization runs. For about 50 iterations the hump of the rod decreases quickly, and the Armijo rule often returns a sufficient improvement already for $t_k = 1$. Thus, the shape converges quickly to a nearly straight joint, however, usually with a wavy boundary, and its probability of failure improves considerably. In later iterations the changes are less visible and the shape converges slowly to a straight joint, the thickness of which depends on the chosen reference point. During the optimization process, the volume decreases only slowly as the wavy boundary gets smoother.

The optimization runs require between 292 and 486 iterations with one exception: for the reference point $r(0.0005)$ the optimization process stops at the maximum of 500 iterations. In all other cases the algorithm terminated in the Armijo rule.

The solutions from the coarse 36×18 grid were used as starting solutions on a refined 71×36 grid with mesh size $m = 0.01\,\mathrm{m}$. Exemplary results for the reference points $r(0.001)$ and $r(0.003)$ are shown by red crosses in Figure 5.15. The respective starting solutions, which are now evaluated on a 71×36 grid, are illustrated by red circles in the same Figure. The results are smoother, see Figure 5.16, but require significantly more computational time. Nevertheless, in this test case setting the coarse grid approximation obtained already considerably good results, which are not dominated by the corresponding results on the finer grid.

Moreover, numerical tests show that shapes representing very thin or thick straight rods, respectively, induce numerical difficulties due to the fixed boundary parts and, particularly, their corner points. Thick rods suffer from overfitted spline representations of their boundaries. Thin rods, on the other hand, lead to distorted finite elements at the spiky corner points.

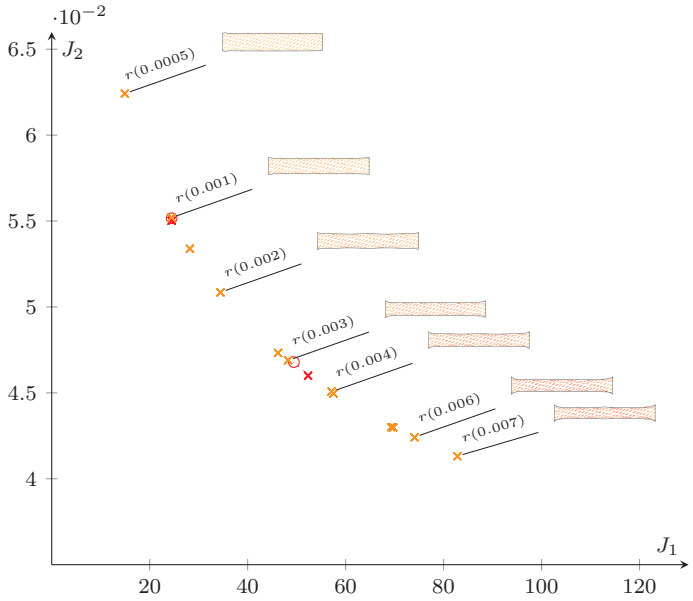


Figure 5.15: Coarse approximation on a 36×18 grid (orange) and two exemplary results from the grid refinement (red).

(a) Solution $\Omega^*_{36\times18}$ evaluated on a 71×36 grid.

(b) Solution $\Omega^*_{71\times36}$.

Figure 5.16: Comparison of the coarse and fine grid solution obtained for reference point $r(0.003)$.

Chapter 6

Efficiency and Reliability of Turbomachinery Components

This chapter focuses on developing a gradient based shape optimization algorithm for turbomachinery components with respect to efficiency and reliability as outlined in the *GivEn* project described in Chapter 1. We start by introducing the chosen test case for a simulation of a low pressure turbine cascade. Next, we present the objectives and their corresponding state equations. Following this, we can formulate the biobjective shape optimization problem, which we aim to solve using the weighted sum method. Subsequently, our attention turns to the implementation details of the optimization algorithm before presenting the results obtained for the chosen test case.

6.1 Model

The optimization aims at enhancing the reliability and efficiency of turbomachinery components. We focus on the simulation of a single turbine stage of a gas turbine, selecting the low-pressure turbine cascade test case T106A for this study, see Hoheisel et al., 1986. For this purpose, our analysis centers on the profile design of the blades. In this test case we consider a set of stationary blades which are designed to remain fixed in position and do not rotate. Stationary blades play an important role in guiding the airflow within the engine towards the rotating blades which are attached to the turbine rotor and spin in order to extract the energy of the airflow. The design of stationary blades significantly influences the efficiency and overall performance of a

gas turbine. In our study, the number of blades is predetermined and not included in the optimization process. Subject of optimization is the profile design, aiming to improve both the efficiency of the surrounding aerodynamic flow and the overall reliability of the component. The profile of one representative blade is given by a cross section and is shown with the surrounding area divided into several sections used for the implementation, see Figure 6.1. On the left side of the boundary the fluid

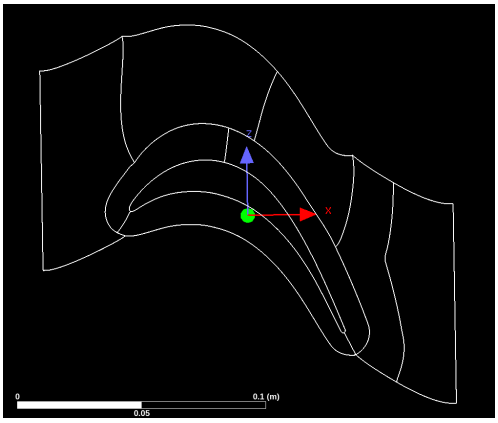


Figure 6.1: Geometry of the T106A profile with surrounding panels.

enters the area, whereas the right boundary indicates where the fluid exists. We call the left and right section *Inlet* and *Outlet*, respectively. The geometry of the T106A configuration is given in pseudo-3D (p3D) for the subsequent calculations, with the profile superimposed twice. The upper and lower boundaries are periodic and the shape is originally given in cylindrical coordinates. As a result, stacking this segment leads to a representation of the turbine cascade. However, to simplify the calculation and realization of shape changes in the optimization algorithm, the cylindrical shape is replaced by a planar shape. Thus, the shape given in p3D is plane in the y-axis.

6.2 Aerodynamic Efficiency

The aerodynamic flow is a key factor in determining the efficiency of turbomachinery components and is evaluated by computational fluid dynamics. For the simulation and computational solution of the aerodynamic state equation and the evaluation of the corresponding objectives we use the tool TRACE (Turbomachinery Research Aerodynamic Computational Environment) developed by the German aerospace center (DLR). In this section we briefly discuss the aerodynamic state problem characterizing

the flow around the component under examination. Based on this formulation several objective functions can be taken into account that relate to efficiency. In Section 6.2.2, we explain the objective to evaluate the aerodynamic efficiency here. This section is based on Backhaus, 2020 and *TRACE User Guide* 2019.

6.2.1 Aerodynamic State

The density, velocity and internal energy of a fluid in a given domain $D \setminus \Omega$ are characterized by its aerodynamic state q given by

$$q = \begin{pmatrix} \rho \\ \rho \mathbf{V} \\ \rho E \end{pmatrix} \tag{6.1}$$

with density ρ, velocity vector $\mathbf{V} = (V_x, V_y, V_z)$ and the total energy E of the fluid. Here, $\Omega \subset \mathbb{R}^3$ is a connected and compact domain representing the component under consideration and $D \subset \mathbb{R}^3$ is a larger, connected and compact that contains Ω. In our setting the aerodynamic state is required to fulfill the compressible Navier-Stokes equations in a rotating frame of reference

$$\frac{\partial q}{\partial t} + \operatorname{div}(F(q)) + S(q) = 0 \tag{6.2}$$

which include the mass, momentum and energy conservation equations. The vector F describing the fluxes is given by

$$F = \begin{pmatrix} \rho \mathbf{V} \\ \rho \mathbf{V} \otimes \mathbf{V} + p \ \mathrm{Id} - \tau \\ \rho \mathbf{V} H_t - \tau \mathbf{V} + \mathbf{Q} \end{pmatrix}. \tag{6.3}$$

Here, $\otimes$ denotes the outer product, i.e., $\mathbf{V} \otimes \mathbf{V} = \mathbf{V}\mathbf{V}^\top$ results in a $(3,3)$-matrix. The pressure is denoted by p and Id is the identity matrix. According to Stokes' law for a Newtonian fluid the viscous stresses are given by

$$\tau_{ij} = 2\mu \left(s_{ij} - \frac{1}{3} \mathrm{tr}\,(s)\, \delta_{ij} \right) \tag{6.4}$$

with

$$s_{ij} = \frac{1}{2} \left(\frac{\partial V_j}{\partial x_i} + \frac{\partial V_i}{\partial x_j} \right) \tag{6.5}$$

where μ denotes the dynamic viscosity and δ_{ij} is the Kronecker delta.

The rothalpy is given by

$$H_t = e + \frac{1}{2}\left(\|\mathbf{V}\|^2 - (|\omega_\mathrm{a}|r)^2\right) + \frac{p}{\rho}.$$

In case of an ideal gas we have internal energy of the gas given by

$$e = \frac{1}{1-\gamma} R_s T \qquad \text{and} \qquad p = \rho R_s T$$

with R_s the specific gas constant, T the temperature and γ the heat capacity ratio. The angular velocity is denoted by ω_a. The heat flux density is calculated by Fourier's law $\mathbf{Q} = -\kappa \nabla T$ with κ the thermal conductivity.

The source term $S(q)$ results from the rotation system containing the Coriolis and centrifugal forces. In TRACE the rotation axis is aligned with the x-axis. Thus, we have

$$S = \begin{pmatrix} 0 \\ 2\rho\,\mathbf{\Omega}_\mathrm{a} \times \mathbf{V} - \rho|\omega_\mathrm{a}|^2 r \mathbf{e}_r \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -2\rho\omega_\mathrm{a} V_z - \rho|\omega_\mathrm{a}|^2 y \\ 2\rho\omega_\mathrm{a} V_y - \rho|\omega_\mathrm{a}|^2 z \\ 0 \end{pmatrix} \tag{6.6}$$

with $\mathbf{\Omega}_\mathrm{a} = (\omega_\mathrm{a}, 0, 0)^\top$ given by the angular velocity. Here, the cross product is denoted by $\times$ and the radial unit vector by $\mathbf{e}_r$. Given that the rotation axis corresponds to the x-axis, the local radius can be determined as $r = \sqrt{y^2 + z^2}$ where y and z denote the Cartesian coordinates along the y- and z-directions, respectively.

Since there is no analytical solution of the Naiver Stokes equation in general, the equations are discretized and solved numerically. Furthermore, given the available computing power, technically relevant flows cannot be discretized finely enough to solve them with the above equations. Following Backhaus, 2020, Favre and Reynolds averages are therefore applied to the above equations. The resulting equations are then called Reynolds-Averaged-Navier-Stokes (RANS) equations. In this process, all variables depending on time are decomposed into a stationary and a fluctuating part where only the stationary part, which corresponds to the averaged quantities, is simulated. The fluctuating part is described with the use of turbulence models. There are different types of turbulence models, see Morsbach, 2017 for a detailed description. Here, we use the Wilcox k–ω model (see Wilcox, 1988) with modifications. Note that here k is the kinetic energy of turbulent fluctuation per unit mass and ω is the specific

dissipation rate. For more details about the discretization and implementation, see Backhaus, 2020 and *TRACE User Guide* 2019.

6.2.2 Efficiency Objective

Numerous evaluation criteria exist in the context of turbomachinery components. For example, typical criteria are the mass flow, total pressure ratio, efficiencies or loss characteristics. To assess the efficiency of the design often the balance of certain parameters between inlet and outlet surfaces of a control volume is considered. The TRACE tool includes a number of objective functions that have already been implemented. We use as objective a criterion called `CoefEntropyRise` which considers the change of the entropy between inlet and outlet given by

$$C_{\Delta s} = \log(p_{\mathrm{t,abs,out,is}}/p_{\mathrm{t,abs,in}}) - \log(p_{\mathrm{t,abs,out}}/p_{\mathrm{t,abs,in}}) \tag{6.7}$$

whereas p_{t} is the total pressure. The subscript *abs* denotes a quantity in absolute frame of reference, i.e., the non-rotating case. The subscripts *is* describes the isentropic property. Moreover, the subscripts *in* and *out* refer to the inlet and outlet locations, respectively. See *TRACE User Guide* 2019 for more details.

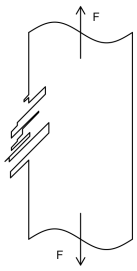
Since the aerodynamic objective function is based on a simulation procedure for the solution of the state equation, each evaluation of the objective function is associated with a significant usage of computer resources. Therefore, we call this objective function *expensive*. Furthermore, not every design can be assessed since a geometrical, mechanical or aerodynamic restriction may cause the simulation to fail. We also note that aerodynamic objective functions in shape optimization may be multimodal, i.e. they may have multiple local minima (or maxima), see Bons et al., 2019.

6.3 Low-Cycle Fatigue

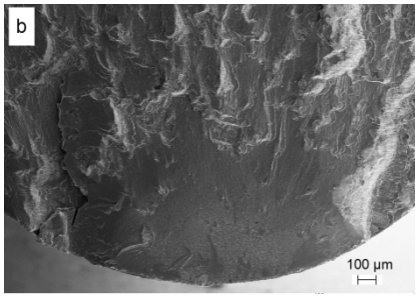
Similar to Chapter 5, our goal is to decrease failure probabilities of mechanical components. But now we consider components made of polycrystalline metal rather than a ceramic material. Moreover, we take cyclic loads into account. The mechanical integrity of a component is understood as the absence of failure. Under cyclic loads mechanical components can fail even if the stresses are far below the ultimate tensile strength of the material, see Radaj and Vormwald, 2007; Rösler, Harders, and Bäker, 2019. This process is called fatigue.

Here, we will consider low-cycle fatigue (LCF) which is characterized by high stress and low amplitude/frequency with plastic strains. LCF models, for example, the switch-on and switch-off process of a gas turbine, or the takeoff and landing process of an aircraft. High cycle fatigue (HCF), on the other hand, is characterized by low stress and high frequency and is not considered here.

We consider a shape made from polycrystalline metal. The degradation process is physically caused by the gliding of linear lattice defects along crystallographic planes of the densest packing towards the so-called slip systems, see Gottstein, 2004. Due to the repetitive load and shear stresses, the lattice defects can reach the surface and form intrusions and extrusions, see Figure 6.2a, where a crack can start to initiate, see Figure 6.2b. Low-cycle fatigue is thus a surface driven failure mechanism.



(a) Intrusions and extrusions caused by cyclic loading F.



(b) Crack initiation at the lower left boudary during a life cycle test for Ni-based speralloy RENE80.

Figure 6.2: Low cycle fatigue is a stress and surface driven failure mechanism.

In a deterministic approach for mechanical integrity, see, e.g., Rösler, Harders, and Bäker, 2019, here understood as the absence of LCF cracks, the numbers of cycles for which the component may be used safely is calculated for every local point at the surface. Then the minimum is taken and some safety margins are added. However, considering the minimum results in a non-differentiable objective function.

We thus follow a different approach in this work. Taking the random nature of LCF crack formation into account we will use a probabilistic approach instead of the deterministic approach to model low-cycle fatigue. Thus, instead of considering the local point with highest stress, we consider an integral over the surface of the component and obtain a differentiable objective that reflects the probability of failure. As a consequence, we can calculate shape sensitivities, see Bittner, 2018 for a throughout analysis. With the shape sensitivities at hand, we can apply gradient based methods

for shape optimization.

We outline the key points of the probabilistic LCF objective below. We start with recalling the mechanical state equation. Note that we consider this PDE already in Section 5.1.2 for the ceramic PoF objective. Then, we describe the deterministic approach for LCF which serves as the basis for the probabilistic model. For the deterministic model for LCF we refer to Gottstein, 2004; Radaj and Vormwald, 2007; Rösler, Harders, and Bäker, 2019. Afterwards, we describe the probabilistic model and conclude with a comparison between the LCF objective and the PoF objective for ceramic material from Chapter 5. Literature on the probabilistic model for LCF that is presented here can be found in Bittner, 2018; Gottschalk and S. Schmitz, 2014; S. Schmitz, 2014; S. Schmitz, Gottschalk, et al., 2013; S. Schmitz, Rollmann, et al., 2013; S. Schmitz, Seibel, et al., 2013, to name a few, and the calculation of probabilistic LCF sensitivities via the adjoint approach is considered in Gottschalk and Saadi, 2019; Hahn, 2021; Saadi, 2021.

6.3.1 Linear Elasticity PDE

We consider a bounded domain $\Omega \subset \mathbb{R}^d$ ($d \in \{2,3\}$ fixed) with a piecewise Lipschitz boundary. The shape Ω represents a mechanical component made of polycrystalline metal in a force free equilibrium. The boundary $\partial\Omega$ is divided into two parts $\partial\Omega = \partial\Omega_D \cup \partial\Omega_N$. The Dirichlet boundary $\partial\Omega_D$ is an open portion of $\partial\Omega$ with non-vanishing surface measure, where the component is fixed. The Neumann boundary $\partial\Omega_N$ is the part of the boundary where the surface load may act.

Now let $f \in L^2(\Omega, \mathbb{R}^d)$ be a volume force like gravity of centrifugal loads and let $g \in L^2(\partial\Omega_N, \mathbb{R}^d)$ be a surface load, e.g., caused by a static gas pressure $p(x) \in \mathbb{R}$ where $x \in \partial\Omega_N$ and $g(x) = -p(x)\hat{n}(x)$ with $\hat{n}$ the outward pointing normal vector field on $\partial\Omega_N$. The displacement field $u \in H^1(\Omega, \mathbb{R}^d)$ caused by these loads is the solution of the following linear elasticity PDE:

$$\begin{aligned} -\operatorname{div}(\sigma(u(x))) &= f(x) && \text{for } x \in \Omega, \\ u(x) &= 0 && \text{for } x \in \partial\Omega_D, \\ \sigma(u(x))\hat{n}(x) &= g(x) && \text{for } x \in \partial\Omega_N. \end{aligned} \tag{6.8}$$

The linearized strain tensor $\varepsilon \in L^2(\Omega, \mathbb{R}^{d\times d})$ is defined by

$$\varepsilon(u(x)) = \frac{1}{2}\left(Du(x) + Du(x)^\top\right) \tag{6.9}$$

with Du the Jacobi matrix of u. We equivalently write $\varepsilon(x) \coloneqq \varepsilon(u(x))$ to describe the strain state at some location $x \in \Omega$. The stress tensor field $\sigma \in L^2(\Omega, \mathbb{R}^{d\times d})$ is defined by

$$\sigma(u(x)) = \lambda \mathrm{tr}(\varepsilon(u(x)))\mathcal{I} + 2\mu\varepsilon(u(x)) \tag{6.10}$$

$$= \lambda \mathrm{div}(u(x))\mathcal{I} + \mu\left(Du(x) + Du(x)^\top\right) \tag{6.11}$$

where $\lambda, \mu > 0$ are the Lamé constants.

Note that we have the same linear elasticity PDE as in Section 5.1.2, but omit the subdivision of $\partial\Omega_N$ with a force-free part $\Omega_{N_{\mathrm{free}}}$. We refer to the linear elasticity PDE (6.8) as the *mechanical state equation*.

6.3.2 Deterministic Approach for LCF

To derive the probabilistic LCF objective we first have a look at the deterministic LCF approach which is based on the Coffin-Mason-Basquin (CMB) equation. According to Rösler, Harders, and Bäker, 2019, the CMB equation is defined by two parts. The first part

$$\varepsilon_a^{\mathrm{el}} = \frac{\sigma_f'}{E_Y}(2N_{i_{\mathrm{det}}})^b \tag{6.12}$$

is the Basquin equation describing the elastic range $\varepsilon_a^{\mathrm{el}}$, where σ_f' is the fatigue strength coefficient, b is the fatigue strength exponent and E_Y is Young's modulus. The second part

$$\varepsilon_a^{\mathrm{pl}} = \varepsilon_f'(2N_{i_{\mathrm{det}}})^c \tag{6.13}$$

is the Coffin-Manson equation describing the plastic range $\varepsilon_a^{\mathrm{pl}}$. Here, ε_f' is the fatigue ductility coefficient and c is the fatigue ductility exponent. The CMB equation is then given by

$$\varepsilon^{\text{el-pl}} = \varepsilon_a^{\mathrm{el}} + \varepsilon_a^{\mathrm{pl}} = \frac{\sigma_f'}{E_Y}(2N_{i_{\mathrm{det}}})^b + \varepsilon_f'(2N_{i_{\mathrm{det}}})^c \tag{CMB}$$

and is used to describe the relationship between strain or stress and the expected number of cycles until crack initiation. Here, $\varepsilon^{\text{el-pl}}$ is called the elastic-plastic strain amplitude. The parameters $\sigma_f' > 0$, $\varepsilon_f' > 0$ and $b < c < 0$ are material constants and are estimated according to test data.

The solution $N_{i_{\mathrm{det}}}$ of the CMB equation is then called the deterministic number of load cycles until crack initiation and can be calculated by the following steps, see, for example, Bittner, 2018; Gottschalk and S. Schmitz, 2014: Given the solution

u of the mechanical state equation (6.8), we have the stress field $\sigma(u)$ given by $\sigma(u) = \lambda \mathrm{div}(u)\mathcal{I} + \mu\left(Du + Du^\top\right)$, $\lambda, \mu > 0$ (see (6.10) and (6.11)). First, we calculate the trace free part of σ given by

$$\sigma' = \mathrm{TF}(\sigma) = \sigma - \frac{1}{3}\mathrm{tr}(\sigma)\mathcal{I}. \tag{TF}$$

We define the amplitude comparison stress as the van Mises stress given by

$$\sigma_v = \mathrm{VM}\left(\sigma'\right) = \sqrt{\frac{3}{2}\sigma' : \sigma'}. \tag{VM}$$

Here, $A : B$ is the Frobenius scalar product defined by $A : B = \mathrm{tr}\left(A^\top B\right)$. Now, the Neuber shake-down (SD) rule (see Neuber, 1961) can be used to determine the elastic-plastic stress amplitude $\sigma^{\text{el-pl}}$ by

$$\sigma_v = \mathrm{SD}\left(\sigma^{\text{el-pl}}\right) = \sqrt{\left(\sigma^{\text{el-pl}}\right)^2 + E_Y \sigma^{\text{el-pl}} \left(\frac{\sigma^{\text{el-pl}}}{K}\right)^{\frac{1}{n'}}} \tag{SD}$$

where E_Y is again Young's modulus, K is the hardening coefficient and n' the hardening exponent. With the Ramberg-Osgood relation (RO) we obtain the elastic-plastic strain amplitude $\varepsilon^{\text{el-pl}}$ based on $\sigma^{\text{el-pl}}$

$$\varepsilon^{\text{el-pl}} = \mathrm{RO}\left(\sigma^{\text{el-pl}}\right) = \frac{\sigma^{\text{el-pl}}}{E_Y} + \left(\frac{\sigma^{\text{el-pl}}}{K}\right)^{\frac{1}{n'}}, \tag{RO}$$

see Ramberg and Osgood, 1943. This can then be used to solve the CMB equation:

$$\varepsilon^{\text{el-pl}} = \mathrm{CMB}(N_{i_{\text{det}}}) = \frac{\sigma'_f}{E_Y}(2N_{i_{\text{det}}})^b + \varepsilon'_f(2N_{i_{\text{det}}})^c. \tag{CMB}$$

Combining these steps, we have the following formula for the deterministic life prediction $N_{i_{\text{det}}} \in \mathbb{R}_+ \cup \{\infty\}$:

$$N_{i_{\text{det}}}\left(\sigma(u(x))\right) = \mathrm{CMB}^{-1} \circ \mathrm{RO} \circ \mathrm{SD}^{-1} \circ \mathrm{VM} \circ \mathrm{TF}(\sigma(u(x))), \quad x \in \partial\Omega. \tag{6.14}$$

6.3.3 Probabilistic Model for LCF

The probabilistic model for LCF is based on a statistical model for crack initiation, see S. Schmitz, Seibel, et al., 2013. This model is based on a Poisson point process, similar to the derivation of the probability of failure objective in Section 5.2.1. We

outline the most important points here and refer to S. Schmitz, Seibel, et al., 2013 for a more detailed description.

An LCF crack initiation is modeled as a random event in space. Let the mechanical component be given by $\Omega \subset \mathbb{R}^d$. We denote the cycle number by $n \geq 0$. The number n is strictly speaking an integer number, but we treat it as a continuous, time-like number for technical reasons. We now assume that the initiation of an LCF crack is defined by a surface location $x \in \partial\Omega$ and a cycle number n. With $B \subset \partial\Omega \times (0, \infty)$ we denote collections of such pairs of location and time (for simplicity, we deliberately suppress some mathematical details, e.g., measurability of B, see again S. Schmitz, Seibel, et al., 2013 for more details). Then we count the number of cracks for a given strain state $\varepsilon(x)$ at some location and time in B and denote it by $N(B, \varepsilon(x))$. We note that $N(B, \varepsilon(x))$ is a random quantity. Whenever we decompose B into a finite collection of pairwise disjoint subsets $B_1, \ldots, B_k$ we obtain

$$N(B, \varepsilon) = \sum_{j=1}^{k} N(B_j, \varepsilon). \tag{6.15}$$

These random counts of crack initiations lead us to a point process. Furthermore, we assume that at one location and point in time, at most one crack can initiate. We have the following assumptions that imply that $N(B, \varepsilon)$ is Poisson distributed, i.e., $N(B, \varepsilon) \sim \mathrm{Po}(\lambda(B, \varepsilon))$, see Kallenberg, 1983; Watanabe, 1964:

- Identification of single cracks: At one location and point in time, at most one crack can occur.
- Local dependence of the load situation: If two surface regions have the same surface area and the same strain state, they have the same statistical properties that a crack will initiate in a given time interval.
- Independence: The initiation of a crack at some surface point at time will not influence the initiation of a crack at another surface location and another time.

Under these assumptions, we have a Poisson point process and the probability for the number of cracks in B is given by

$$P(N(B, \varepsilon) = k) = \mathrm{e}^{-\lambda(B,\varepsilon)} \frac{\lambda(B, \varepsilon)^k}{k!}, \qquad k = 0, 1, 2, \ldots \tag{6.16}$$

with $\lambda(B, \varepsilon)$ being the intensity parameter of the Poisson distribution. A suitable

model for $\lambda(B, \varepsilon)$ is given by

$$\lambda(B, \varepsilon) = \int_B \rho(n, \varepsilon) \, \mathrm{d}A\mathrm{d}n \tag{6.17}$$

where $\rho(n, \varepsilon)$ is the crack formation intensity function and $\mathrm{d}A$ the surface volume measure. Since we are interested in the situation where the whole component is crack free, we consider the survival probability up to time n. With N_i we denote the (random) time of initiation of the first crack. The survival probability $S_{N_i}(n)$ up to time n is then given by

$$\begin{aligned} S_{N_i}(n) &= P(N(\partial\Omega \times (0, n], \varepsilon) = 0) \\ &= \exp\left\{ - \int_0^n \int_{\partial\Omega} \rho(n, \varepsilon) \, \mathrm{d}A\mathrm{d}n \right\} \end{aligned} \tag{6.18}$$

and the probability of failure is given by

$$\begin{aligned} F_{N_i}(n) &= 1 - S_{N_i}(n) \\ &= 1 - \exp\left\{ - \int_0^n \int_{\partial\Omega} \rho(n, \varepsilon) \, \mathrm{d}A\mathrm{d}n \right\}. \end{aligned} \tag{6.19}$$

Furthermore, the hazard rate function is given by

$$\begin{aligned} h_{N_i}(n) &= \lim_{\Delta n \searrow 0} \frac{1}{\Delta n} P\left(N_i \in (n, n + \Delta n] \,|\, N_i > n\right) \\ &= \lim_{\Delta n \searrow 0} \frac{1}{\Delta n} \frac{F_{N_i}(n + \Delta n) - F_{N_i}(n)}{S_{N_i}(n)} \\ &= \int_{\partial\Omega} \rho(n, \varepsilon) \, \mathrm{d}A \end{aligned} \tag{6.20}$$

which describes the probability of failure in the next moment given that the component survived up to time n. This describes a measure of risk.

Now we need a model for the crack formation intensity measure $\rho(n, \varepsilon)$. Based on the solution of the (CMB) and the Weibull approach, see Weibull, 1939, in S. Schmitz, Seibel, et al., 2013 the following model is proposed

$$\rho(n, \varepsilon(x)) = \frac{m}{N_{i_{\mathrm{det}}}(\sigma(u(x))} \left(\frac{n}{N_{i_{\mathrm{det}}}(\sigma(u(x)))} \right)^{m-1} \tag{6.21}$$

with some shape parameter m and scale parameter $N_{i_{\text{det}}}(\sigma(x))$, $x \in \partial\Omega$ as defined in the (CMB) equation. Note that the stress field σ can be calculated by the strain field ε, see (6.10). The value $N_{i_{\text{det}}}(x)$ can be interpreted as the deterministic number of life cycles for every point $x \in \partial\Omega$. Inserting (6.21) in (6.19) we have the probability of failure for LCF crack initiation in the time interval $(0, n]$ given by

$$\begin{aligned} F_{N_i}(n) &= 1 - \exp\left\{ -\int_0^n \int_{\partial\Omega} \frac{m}{N_{i_{\text{det}}}} \left(\frac{n}{N_{i_{\text{det}}}} \right)^{m-1} \mathrm{d}A\mathrm{d}n \right\} \\ &= 1 - \exp\left\{ -\int_{\partial\Omega} \left(\frac{n}{N_{i_{\text{det}}}} \right)^{m} \mathrm{d}A\mathrm{d}n \right\}. \end{aligned} \tag{6.22}$$

We call $m \geq 1$ the Weibull shape parameter. We assume that m is independent of the strain field ε. The shape parameter m determines the scatter of the distribution. Small values correspond to large scatter and $m \to \infty$ is the deterministic limit. Note that values $0 < m < 1$ are not realistic for fatigue as this would imply that the hazards decrease over time.

Now we can define our objective value for LCF as

$$J_{\text{LCF}}(\Omega, u(\Omega)) = \int_{\partial\Omega} \left(\frac{1}{N_{i_{\text{det}}}(\sigma(u(x)))} \right)^{m} \mathrm{d}A. \tag{6.23}$$

The probability of failure $\text{PoF}(n)$, i.e., the probability of an LCF crack after n load cycles, is then given by

$$\text{PoF}(n) = F_{N_i}(n) = 1 - \mathrm{e}^{-n^m J_{\text{LCF}}(\Omega, u(\Omega))}. \tag{6.24}$$

We note that minimizing J_{LCF} is equivalent to minimizing $\text{PoF}(n)$.

For the calculation of J_{LCF} by (6.23) we calculate $N_{i_{\text{det}}}(\sigma(u(x))$ by formula (6.14), but have to adjust the material constants σ'_f and ε'_f and its units in the (CMB) equation. Additionally, the constants σ'_f and ε'_f have to be calibrated to the probabilistic model because the solution of the deterministic (CMB) equation usually belongs to the 50%-quantile curve. For the solution of the Weibull model we need the parameter corresponding to the $1-\frac{1}{\mathrm{e}} \approx 63\%$-quantile curve (see $n = N_{i_{\text{det}}}$ in (6.22)). Furthermore, we have to take the statistical size effect into account. The size effect states that given the same strain state larger components have the tendency to fail earlier than smaller ones. We skip the details on the calibration and refer to Gottschalk and Saadi, 2019;

S. Schmitz, Seibel, et al., 2013.

Note that the LCF objective and the probability of failure (PoF) objective introduced for ceramic materials in Chapter 5 both aim at a minimization of failure probabilities as an objective, and both are modeled on the basis of a Poisson point process. However, for the ceramic PoF we consider a one-time application of load and have a volume driven failure mechanism, whereas for the LCF objective repeated mechanical loading is applied and we have a surface driven failure mechanism. With a suitable finite element analysis, derivative information can be computed for both objectives. However, for the LCF objective we need stronger assumptions on the regularity of the solutions u of the mechanical state equation (6.8) since, e.g., for the calculation of the stress σ we consider first derivatives of u. If we only have solutions $u \in L^2(\Omega, \mathbb{R}^d)$, the restriction to the boundary $\partial\Omega$ is not necessarily well defined, see, for example, Gottschalk and S. Schmitz, 2014.

6.4 Biobjective Formulation

In the previous sections we have introduced the test case along with its objectives and corresponding state equations. The goal is to improve both aerodynamic efficiency and mechanical reliability of a turbine blade by optimizing its shape denoted as Ω. To achieve this, we formulate a biobjective shape optimization problem which we aim to solve by the implementation of a weighted sum algorithm, see Section 2.2.1. The first objective, denoted as J_{Eff}, quantifies the change in entropy between the fluid's inlet and outlet surfaces serving as a measure for efficiency. The second objective J_{LCF} is correlated with the component's failure probability under cyclic loading conditions and thus, takes the reliability into account. Both objectives rely on an associated state equation that needs to be solved by finite element methods. The surface load of the mechanical state equation (6.8) is determined by the pressure values from the aerodynamic state equation (6.2). Note that both objectives are formulated to be minimized in order to increase efficiency and reliability. In summary, we have the biobjective shape optimization problem given by

$$
\begin{aligned}
&\min \quad J_{\text{LCF}}(\Omega) = \text{LCF}(\Omega, u_{\text{mech}}(\Omega)) && (6.23)\\
&\min \quad J_{\text{Eff}}(\Omega) = \texttt{CoefEntropyRise}(\Omega, u_{\text{aero}}(\Omega)) && (6.7)\\
&\text{s. t.} \quad u_{\text{mech}}(\Omega) \text{ solves the state equation (6.8) on } \Omega, && (6.25)\\
&\qquad\; u_{\text{aero}}(\Omega) \text{ solves the state equation (6.2) on } D \setminus \Omega,\\
&\qquad\; \Omega \in \mathcal{O}^{\text{ad}}.
\end{aligned}
$$

The weighted sum scalarization is then given by

$$\mathcal{J}(\Omega) = \omega_{\mathrm{LCF}} \cdot J_{\mathrm{LCF}}(\Omega) + \omega_{\mathrm{Eff}} \cdot J_{\mathrm{Eff}}(\Omega) \tag{6.26}$$

with fixed weights such that $\omega_{\mathrm{LCF}}, \omega_{\mathrm{Eff}} \in (0,1)$ and $\omega_{\mathrm{LCF}} + \omega_{\mathrm{Eff}} = 1$. We will discuss the specific choice of weights in Section 6.6. The aim is to obtain different solutions with different weightings and thus to achieve an approximation of the Pareto front.

6.5 Implementation

The gradient based optimization algorithm designed for solving problem (6.25) is composed of several different elements, which we briefly delineate below and then put together in a coupling routine. In a nutshell, the implementation comprises four key elements: firstly, the utilization of the TRACE software package from DLR to handle the efficiency objective; secondly, an in-house implementation by Dr. Camilla Hahn to compute the LCF objective and sensitivities, see Hahn, 2021; thirdly, a specialized software package crafted by Dr. Daniel Luft for computing Steklov Poincaré gradients within the TRACE framework, see Luft, 2021, and lastly a coupling routine putting everything together and adding an iterative solver to solve the weighted sum scalarizations (6.26) of problem (6.25) implemented as part of this thesis. The code was developed in close cooperation with Dr. Jan Backhaus, Dr. Daniel Luft, Dr. Camilla Hahn and Prof. Dr. Matthias Bolten as part of the GivEn project, see Chapter 1.

6.5.1 Efficiency and Exterior Discretization

As stated in Section 6.2 we use the TRACE tool for the calculation of the efficiency objective and sensitivities, see *TRACE User Guide* 2019. The TRACE software package (Turbomachinery Research Aerodynamic Computational Environment) consists of a CFD (Computational Fluid Dynamics) code developed at the DLR (German Aerospace Center) in order to analyze turbomachinery flows. It is the primary method for simulation of internal flows within the DLR. Outside DLR, universities, research institutions, MTU Aero Engines, and Siemens Energy use TRACE for scientific analysis and industrial design optimization of turbomachinery components.

Here, we use TRACE version 9.2 for the optimization purposes within the GivEn project. In particular, we use *adjointTRACE*, the discrete adjoint solver of the turbomachinery simulation suite TRACE. The idea of adjoint methods to calculate

gradients is shortly outlined in Section 4.1. In adjointTRACE the gradients are computed using algorithmic differentiation in reverse mode. For the construction of the discrete adjoint solver and further information, see Backhaus, A. Schmitz, et al., 2017; Becker, Heitkamp, and Kügeler, 2010; Sagebaum et al., 2017.

In the following, we distinguish between the TRACE's primal solver and its adjoint solver. The initial stage involves solving the primal problem (RANS equations, as outlined in Section 6.2). Subsequently, several objectives can be evaluated. Here, we use a coefficient that quantifies the entropy rise ratio between surface inlet and outlet, denoted as *CoefEntropyRise*, see Section 6.2.2. Afterwards, we can employ the adjoint solver to calculate sensitivities of the associated objective.

TRACE can work with the CGNS file format (computational fluid dynamics general notation system). CGNS is a universal, portable, and extensible standard for storing and retrieving CFD analysis data. This means that it does not require one specific software or platform. It can be used on various computing systems and architectures. This data includes information about the computational grid, simulation parameters, boundary conditions, and the results of the simulation such as velocity fields and pressure distributions. Therefore, the results of TRACE are given in the CGNS file format. We also use this file format to obtain the boundary grid points and pressure values that enter the coupling in the optimization algorithm.

In contrast to the mechanical state equation, the aerodynamic state equation takes the surrounding area of the component into account. To simulate the flow around the component, the surrounding area is discretized using finite elements. Refer to Figure 6.3 for a detailed view of the leading edge of the considered turbine blade showing an structured grid consisting of multiple parts. The grid exhibits finer resolutions towards the boundary of the blade.

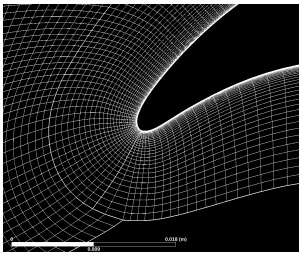


Figure 6.3: TRACE mesh around the leading edge of the T106A profile.

Note that the computations are performed on a pseudo 3D version of the component, i.e. the 2D profile is stacked twice on the top of each other.

6.5.2 Reliability and Interior Discretization

The implementation of the LCF solver was done by Hahn, 2021 in Python 3 within the framework of the *GivEn* project. The application to the optimization problem considered here is described in the following.

Different to the aerodynamic state equation, here the interior of the component has to be discretized. We use the same finite element discretization as stated in Section 5.5.1. Since the p3D planar shape of the T106A component is flat in one dimension, we simply omit this dimension an obtain a 2D version. For this purpose, we again define a rectangle that completely encloses a 2D version of the component. We choose $[-0.02, 0.1] \times [5.92, 6.04]$ that fits the starting shape of the T106A and offers enough space for deformations, see Figure 6.4. This rectangle is discretized by a regular triangular grid with 400 grid points in the x- and y-direction, respectively. We get the boundary points of the current considered component from the CGNS-file. Given these points, we define two splines, defining a function of the lower and the upper part of the component boundary. Here, we chose monotonic cubic interpolation of the given boundary points from the CGNS file by the *scipy.interpolate.PchipInterpolator* class. The surface forces of the Neumann condition in the linear elasticity PDE (see

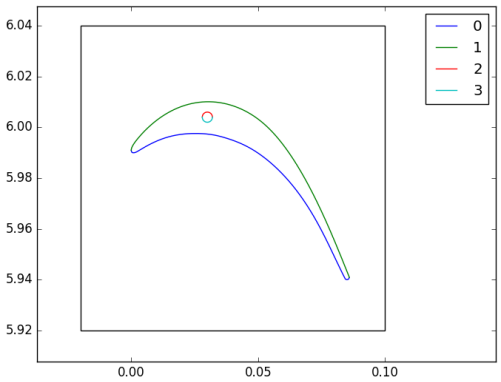


Figure 6.4: Boundary splines for the LCF solver.

Section 6.3.1) are given by the pressure values on the boundary of the component. These values are computed by TRACE in the aerodynamic state equation (see Section 6.2.1) and therefore saved for the current shape in the corresponding CGNS file. We also add a small circle in the inner part of the boundary for the Dirichlet boundary condition, see Figure 6.4, so that we have a unique solution for the mechanical state equation.

As described in Section 5.5.1 and Hahn, 2021 the grid points closest to the boundary of the component are shifted such that they lie on the boundary. The adapted grid representing the component, see Figure 6.5, is then used for the further computations. Note that all calculations of the LCF solver are performed in 2D.

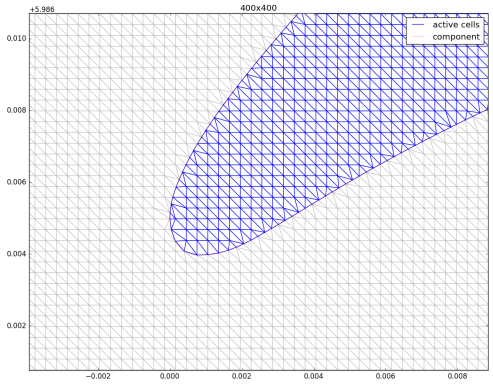


Figure 6.5: LCF mesh in blue at the leading edge of the T106A profile.

6.5.3 Steklov-Poincaré Gradients

A further important element for the optimization routine is the TRASOR (TRACE Shape Optimization Routine) software package implemented by Luft, 2021 within the framework of the *GivEn* project, see also Backhaus, Bolten, et al., 2021. It is used to translate the TRACE grid to a *FEniCS* mesh, which can then be used to calculate shape gradient representations using the Steklov-Poincaré metrics. FEniCS is a Python based finite element computational software which is able to solve differential equations based on weak formulations, see Alnæs et al., 2015. For this purpose, among others, many sub-modules are used, such as, for example, *Automated Finite Element Computing (DOLFIN)* (see Logg and Wells, 2010) or *PETSc* as a linear algebra back end (see Balay et al., 2019).

Since the FEniCS 2017.2.0 version used here is not able to handle finite element grids consisting of hexahedral and quadrilateral elements, we have to translate the mesh given by TRACE to a mesh consisting of conforming tetrahedral and triangular elements corresponding to the existing structure of the TRACE mesh. We call this mesh *FEniCS-aero mesh*. The conversion process takes place in several steps. The individual steps including the data formats for the conversion from the TRACE mesh

to the FEniCS-aero mesh are illustrated in Figure 6.6. Here, *POST* is a command of the TRACE tool used to extract data from the CGNS files and *meshio* (see Schlömer, 2019) is a Python package that can process and convert various mesh formats.

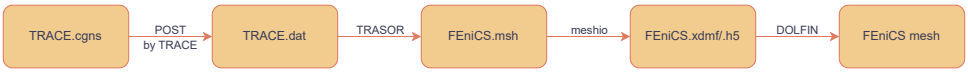


Figure 6.6: Conversion from TRACE mesh to FEniCS mesh.

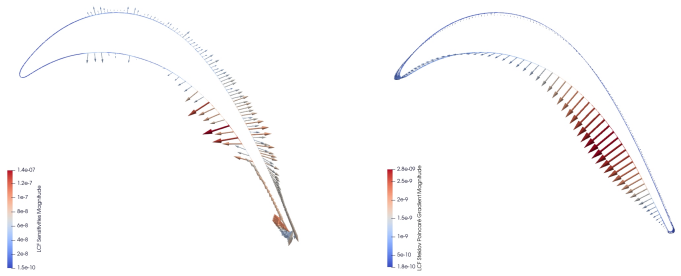
Since the calculated raw sensitivities are generally not smooth enough, we would quickly end up with infeasible shapes if we would use them in an iterative line search algorithm (Algorithm 1). Therefore, we would not be able to achieve any improvements in the objective functions with these sensitivities as search direction. Instead of using the raw sensitivity, the Steklov-Poincaré gradients allow us to smooth the shape change, making them more suitable for shape optimization, see Schulz and Siebenborn, 2016; Schulz, Siebenborn, and Welker, 2016. The Steklov-Poincaré gradients are given by the solution of a linear elasticity system

$$\begin{aligned} \int_{\Omega_{\text{ext}}} \sigma\left(\nabla^{\text{StP}} J(\Omega_{\text{ext}})\right) : \varepsilon(U)\mathrm{dx} = DJ(\Omega_{\text{ext}})[U] \quad U \in H_0^1(\Omega_{\text{ext}}, \mathbb{R}^d) \\ \nabla^{\text{StP}} J(\Omega_{\text{ext}}) = 0 \quad \text{on } \Gamma_{\text{Inlet/Outlet}} \end{aligned} \tag{6.27}$$

with $\lambda = 0$ and $\mu \geq 0$ where Ω_{ext} is the external computational domain given by the FEniCS-aero mesh and $\sigma(U) = \lambda \text{trace}\left(\varepsilon(U)\right) I + 2\mu\varepsilon(U)$. Note that the Steklov-Poincaré metric is not in direct relation to the state equation of the objective function J currently under consideration. In Figure 6.7 the negative sensitivities and the negative Steklov-Poincaré gradient on the boundary of the T106A profile for the LCF objective are shown. The TRASOR package in the context of the GivEn project has already been mentioned in Backhaus, Bolten, et al., 2021. For further details, we refer to Luft, 2021.

6.5.4 Coupling

We want to implement a gradient based iterative solver for the biobjective optimization problem stated in (6.25). For this purpose, the just mentioned building blocks will be combined. We use the TRACE tool for the calculations of the aerodynamic part and we use the LCF solver to assess the reliability. The TRASOR tool is used as basis for the implementation and also used to calculate Streklov-Poincaré gradients.



(a) (b)

Figure 6.7: Negative LCF sensitivities and negative LCF Steklov-Poincaré gradient.

The main task of the coupling is to extend the algorithm already developed in the TRASOR package to handle more than one objective function by integrating the LCF objective functional. In addition, we implement the iterative line search algorithm, see Algorithm 3, for the weighted sum scalarization of (6.25) in this context. Note that since the TRASOR package requires Python 2.7 we use Python 2.7.12 for the coupling code whereas the LCF code is run by Python 3.5.2.

The algorithm is composed of multiple parts. It is outlined in Figure 6.10 in a flow chart to provide an initial overview of its fundamental components. The flow chart is based on the Business Process Model and Notation (BPMN) in a slightly modified form (OMG, 2011). In this flow chart three vertical lanes (called swim lanes), labeled as efficiency, coupling and reliability, are used to represent relevant aspects of the process. On the left side we have the efficiency swim lane with the elements of the process given by the TRACE tool. The right swim lane represents the LCF code for the mechanical reliability. In the middle, coupling relevant elements are shown. The first blue block includes the computations performed for the computation of the objective values. Here, it is illustrated that the pressure values required for the LCF objective are calculated by TRACE. The second blue block comprises the gathering of gradient information including the Steklov-Poincaré computation. Next, the stopping conditions are checked. Besides the calculation of the objective values and gradients, we have to determine a search direction and step length. The search direction is given by the weighted sum of the Steklov-Poincaré gradients of the objectives J_{Eff} and J_{LCF}, whereas the step size will be calculated by the Armijo rule. Thus, if no stopping condition is satisfied, a shape update follows. Since it is an iterative algorithm that computes different evolving shapes, we have to iteratively calculate objective values

and assemble gradient information. The gradients are calculated for the current shape only if the shape is the initial shape or the current step size satisfies the Armijo rule. Otherwise, the current shape candidate is discarded and the step size will be updated to compute a new shape candidate. Since this illustration shows only a simplified flow chart of the algorithm, we will now focus on some details that are of greater importance for the implementation.

To determine a search direction for the line search as a weighted sum of the gradients we have to define the gradients on the same domain. Since the aerodynamic objective is evaluated on an external domain and the LCF objective is evaluated on an internal domain, they do not share the same computational domain and only intersect at the boundary of the component. Since the Steklov-Poincaré metric is already implemented for the external domain by the TRASOR package, we want to define the LCF sensitivities also on the external domain in order to calculate the LCF Steklov-Poincaré gradient there. Note that it would also be possible to perform the calculation of the Steklov-Poincaré gradients on the LCF mesh. However, since we need the gradients in the same domain for the further computations anyways, this would only require unnecessary additional implementations. To represent the LCF sensitivities on the external domain, we again use FEniCS as an interface. Besides the TRACE mesh from the TRACE tool, the FEniCS-aero mesh from the TRASOR package and the LCF mesh of the LCF solver, we now define another mesh for the boundary points of the LCF mesh. We call this mesh *FEniCS-mech mesh*, which is a mesh in FEniCS format with the boundary points of the LCF mesh as nodes. Since the LCF mesh is in 2D we extend it to pseudo-3D by matching the third dimension of the TRACE/FEniCS-aero mesh by simply adding the same points as a second level matching the two levels of the TRACE mesh. The relation between the different meshes is depicted in Figure 6.8. With this additional mesh it is now possible to interpolate the LCF sensitivities onto the FEniCS-aero mesh that represents the external domain. Conversely, we can display the flow solution, especially the pressure values, resulting from (6.2) on boundary points of the LCF mesh to set up the mechanical state equation (6.8). Given the sensitivities both on the same mesh, we can solve for the Steklov-Poincaré gradients and combine them in a weighted sum. The individual steps required for this are summarized in Figure 6.9.

Given the search direction as a displacement vector, the next step is to determine the step size. To find a suitable step size for the initial step in every iteration we scale the displacement vector beforehand. In this way, we avoid unnecessary step sizes that

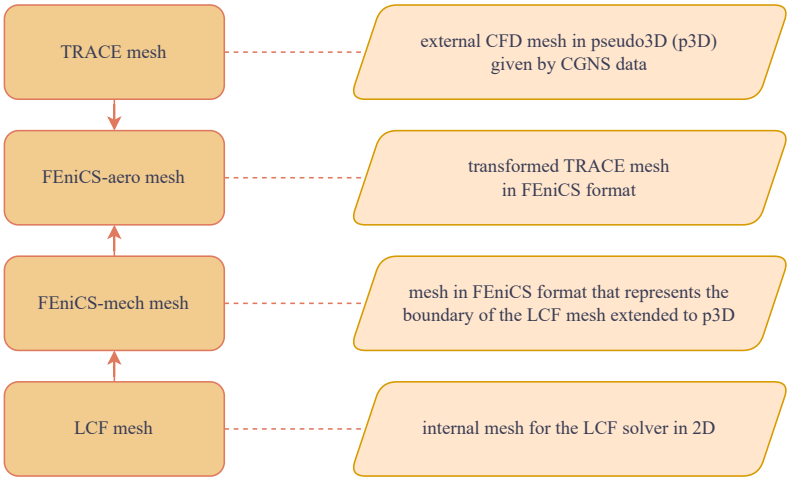


Figure 6.8: Relation between the different meshes.

are too large and lead to inadmissible shapes and simulation failures. In addition, initial step sizes that are too small for a significant change in the shape are scaled up so as not to preclude further improvement. For that purpose, we calculate the ℓ_2-norm of the displacement vector D at every discretized surface point X_{sur} of the shape X:

$$c_{\max} = \max_{x \in X_{\text{sur}}} \|D(x)\|_{\ell_2}. \tag{6.28}$$

Then we scale the displacement field in that way that the longest vector is scaled to the length of one:

$$D_{\vec{n}} = \frac{1}{c_{\max}} \cdot D. \tag{6.29}$$

Thus, we have a normalized displacement field where every single vector has a length of less or equal to one. Now, to avoid unsuitable steps we determine a value $c_{\text{init}} > 0$ depending on the shape optimization problem with which the normalized displacement field is scaled again:

$$D_{\text{scaled}} = c_{\text{init}} \cdot D_{\vec{n}}. \tag{6.30}$$

All in all, the process steps including the different meshes are outlined in Algorithm 6. The objective is to minimize a weighted sum scalarization of J_{LCF} and J_{Eff} with respect to the design variable Ω. We start in Line 1 with selecting certain parameters, such as the parameters for the Armijo rule $\beta \in (0,1)$ and $\sigma \in (0,1)$ and the the weights for the weighted sum scalarization $\omega_1 \in (0,1)$ and $\omega_2 = 1 - \omega_1$. Additionally, we set the iteration counter k to zero. Next, in Lines 2 to 9 we calculate

objective values of the initial shape $\Omega^{(k)}$. We first run TRACE primal for $\Omega^{(k)}$ in Line 2 in order to solve the aerodynamic state equation (6.2). Lines 3 to 6 are for constructing the necessary meshes. We are then able to interpolate the pressure values to the LCF grid in Line 7. After that we can run the LCF solver in Line 8. In the following lines the sensitivities are calculated, followed by the computation of the Steklov-Poincaré gradients $\nabla^{\mathrm{StP}} J_{\mathrm{Eff}}(\Omega^{(k)})$ and $\nabla^{\mathrm{StP}} J_{\mathrm{LCF}}(\Omega^{(k)})$. Based on the Steklov-Poincaré gradients we construct the search direction given by the negative Steklov-Poincaré gradient of the weighted sum scalarization

$$U^{(k)} = -\left(\omega_{\mathrm{LCF}} \cdot \nabla^{\mathrm{StP}} J_{\mathrm{LCF}}(\Omega^{(k)}) + \omega_{\mathrm{Eff}} \cdot \nabla^{\mathrm{StP}} J_{\mathrm{Eff}}(\Omega^{(k)})\right). \tag{6.31}$$

In the while-loop in Line 10 we search for the next shape and increase the iteration counter k by one. In Line 26 a candidate shape is computed by applying the deformation on the current shape by using the TRACE tool. In order to calculate the objective values for the new candidate shape, we have to run Line 2 to 9 again. Then we can check the Armijo rule

$$\mathcal{J}(\Omega^{(k)}) < \mathcal{J}(\Omega^{(k-1)}) \; - \sigma \cdot \beta^{(\ell)} \left(U^{(k-1)}\right)^{\top} U^{(k-1)}_{\mathrm{scaled}}. \tag{6.32}$$

If the Armijo rule is satisfied, we have the new shape and calculate the gradients and search direction for the next iteration. Otherwise, we try the next step length. The algorithm terminates when a maximal number of iterations ($k_{\max}$) is reached, see Line 10, or when a maximal number of steps was tried ($\ell_{\max}$), see Line 18. Moreover, additional stopping conditions based on the gradient information can be examined in Line 18. For example, it can be checked if the norm of the weighted sum gradient is lower than a certain threshold. At the moment, we only stop if a maximal number of iterations is reached or if no sufficiently large step can be calculated.

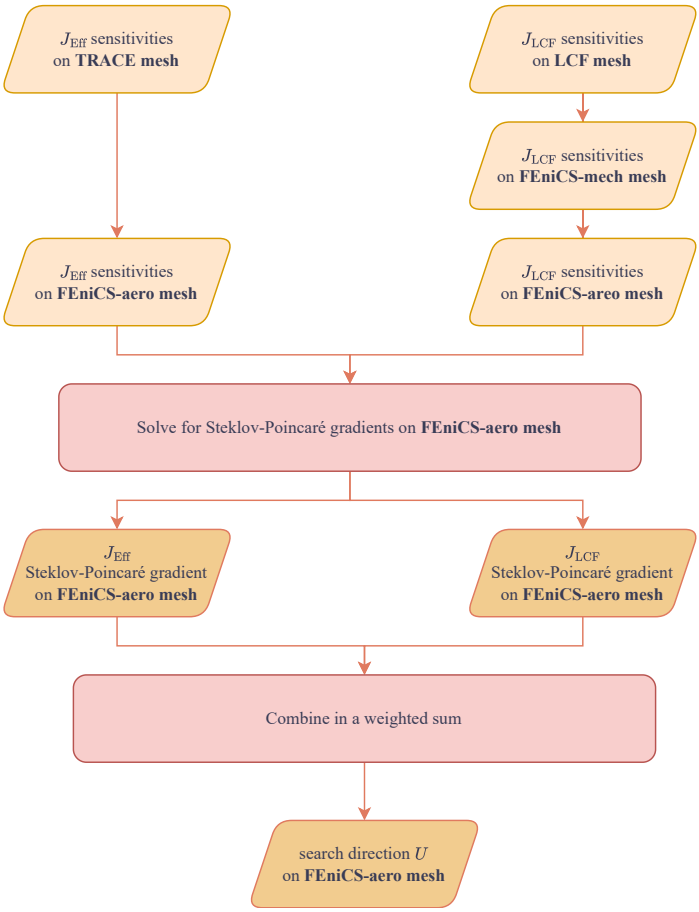


Figure 6.9: Steps to determine the biobjective search direction.

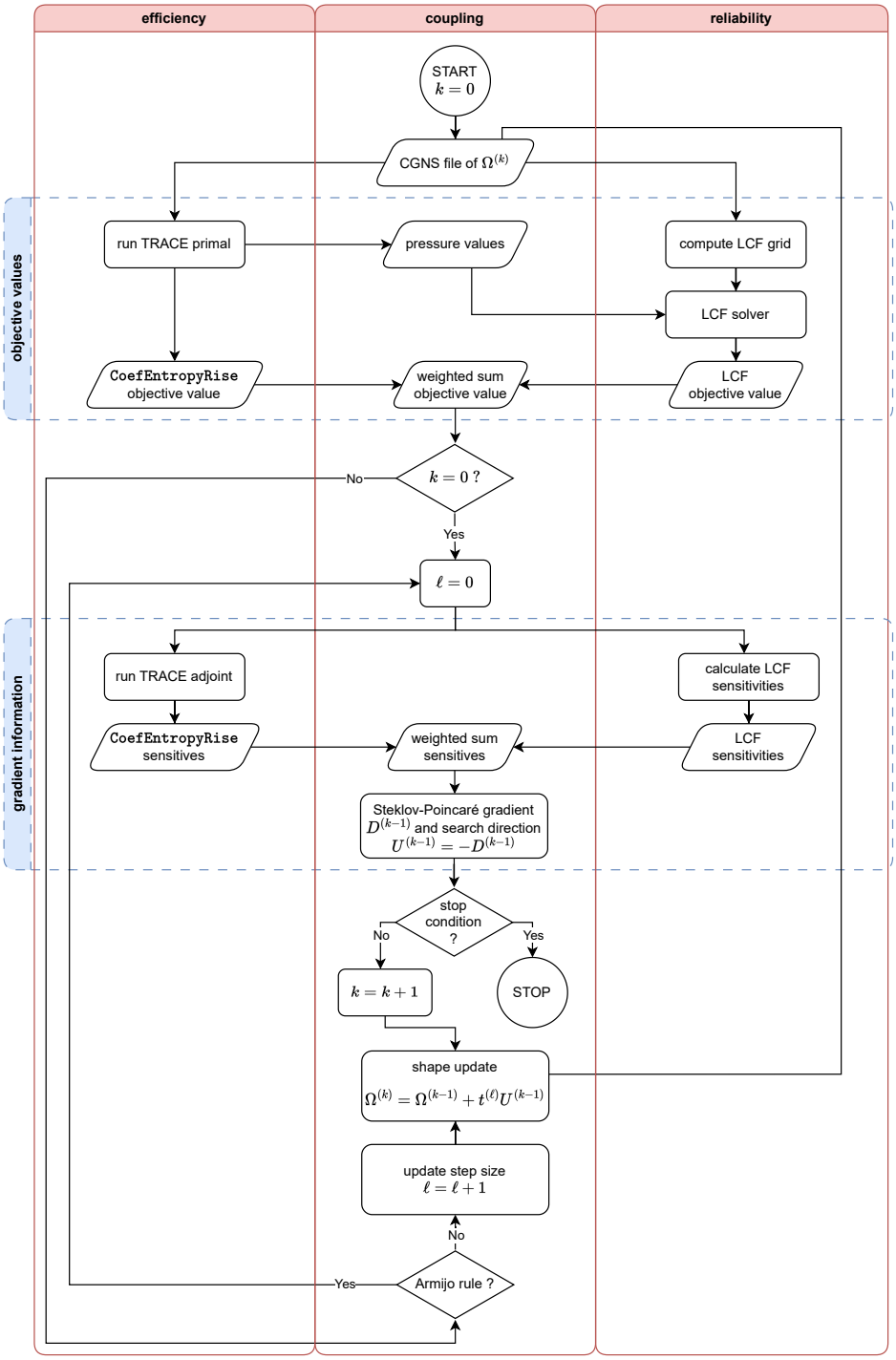


Figure 6.10: Illustration of the coupling structure.

1 Set initial parameters, $k = 0$;
2 Run TRACE primal on $\Omega^{(k)}$ (solve aerodynamic state equation) ;
3 Extract boundary points from CGNS file;
4 Construct FEniCS-aero mesh based on TRACE mesh;
5 Construct LCF mesh;
6 Construct FEniCS-mech mesh based on LCF boundary points;
7 Interpolate pressure values p to LCF boundary points via FEniCS meshes;
8 Run LCF solver;
9 Calculate weighted sum objective value $\mathcal{J}(\Omega^{(k)})$;
10 **while** $k \leq k_{\max}$ *and* $\ell \leq \ell_{\max}$ **do**
11 Calculate TRACE sensitivities using adjointTRACE ;
12 Assemble TRACE sensistivities as FEniCS vector function;
13 Calculate LCF sensitivities;
14 Interpolate LCF sensitivities to FEniCS mesh;
15 Solve for Steklov-Poincaré gradients;
16 Construct search direction on FEniCS mesh via (6.31);
17 Calculate norms;
18 **if** *stopping condition satisfied* **then**
19 STOP;
20 **else**
21 Transform search direction $U^{(k)}$ from FEniCS mesh to TRACE mesh;
22 Prescale search direction via (6.28) to (6.30);
23 Set $k = k + 1$ and $\ell = 0$;
24 **while** $\ell \leq \ell_{\max}$ **do**
25 Calculate step length $t^{(\ell)} = \beta^{\ell}$;
26 Update geometry $\Omega^{(k)} \leftarrow \Omega^{(k-1)} + t^{(\ell)} \cdot U_{\text{scaled}}^{(k-1)}$ using TRACE;
27 Run Lines 2 to 9 for $\Omega^{(k)}$;
28 **if** $\mathcal{J}(\Omega^{(k)}) < \mathcal{J}(\Omega^{(k-1)})$ + *Armijo term* **then**
29 BREAK (go to Line 10);
30 **else**
31 $\ell = \ell + 1$
32 **end**
33 **end**
34 **end**
35 **end**

Algorithm 6: Coupling routine.

6.6 Results

The initial shape is defined by the T106A configuration, with corresponding objective values outlined in Table 6.1. Having a look at the initial objective values, it becomes evident that there is a significant discrepancy in the order of magnitude of the objective values for J_{Eff} and J_{LCF}. While the value for J_{LCF} is relatively small, the value for J_{Eff} exhibits considerable magnitude. This must be taken into account when selecting the weights for the weighted sum scalarization (6.26). Recall that the weights ω_{Eff} and ω_{LCF} are selected such that their sum equals 1. Thus, we will specify in the following only the weight ω_{Eff}. The other weight is then calculated by $\omega_{\text{LCF}} = 1 - \omega_{\text{Eff}}$.

Iteration	J_{Eff}	J_{LCF}
0	564354.511214	$1.6318045436642406 \cdot 10^{-8}$

Table 6.1: T106A objective values.

Since we have no explicit geometrical restrictions in the implementation, it is possible that the algorithm may seek to assess infeasible shapes. This could occur, for example, if boundaries of the component start to overlap or if the shape exceeds the permissible domain. In the case of shapes that can not be evaluated, the algorithm proceeds with the next Armijo iteration. The Armijo iterations are limited by $\ell_{\max} = 75$ and the overall iteration loop is limited by $k_{\max} = 250$, see Algorithm 6. The initial step size scaling is set to $c_{\text{init}} = 0.001$, see (6.30). The values for the Armijo rule are set to $\beta = 0.9$ and $\sigma = 0$, see (6.32). Normally, it is recommend to choose $\sigma > 0$ to achieve a sufficient improvement. However, for the beginning we search for any improvement in the objective value of the weighted sum scalarization, i.e., we only check if $\mathcal{J}(\Omega^{(k)}) < \mathcal{J}(\Omega^{(k-1)})$ is true.

Single-objective optimization First of all, we test what happens when we choose the weights $\omega_{\text{Eff}} \in \{0, 1\}$ to get an idea of where the two individual objectives are aiming at. With this choice of weights, it is unlikely that the algorithm converges to a local minimum since the algorithm probably approaches infeasible shapes. In addition, we only minimize one of the two individual objectives without taking the other objective into account. In Table 6.2 we provide the objective values of the iteration history for the weights $\omega_{\text{Eff}} = 0$ and $\omega_{\text{Eff}} = 1$. In both runs it can be seen that while one objective improves, the other objective deteriorates. Furthermore, the changes of J_{Eff} are relatively small compared to its initial value. In contrast, the changes of J_{LCF} objective are relatively strong compared to its initial value, as the

values more than double or halve. In Figure 6.11 the objective values of the iteration history for both runs are displayed.

The run with $\omega_{\text{Eff}} = 0$ terminated after 9 iterations due to failure by TRACE performing further shape updates. The run with $\omega_{\text{Eff}} = 1$ terminated after 4 iterations which was caused by overlapping boundaries. The initial shape of the T106A configuration and both final shapes are displayed in Figure 6.12.

Dichotomic search scheme To determine further weights, we follow a dichotomic search scheme, see, for example, Aneja and Nair, 1979; Cohon, 1978; Przybylski, Klamroth, and Lacour, 2019. Given two points in the objective space, $x = (x_{\text{Eff}}, x_{\text{LCF}})$ and $y = (y_{\text{Eff}}, y_{\text{LCF}})$, a new weight can be calculated by

$$\begin{pmatrix} \omega_{\text{Eff}} \\ \omega_{\text{LCF}} \end{pmatrix} = \frac{1}{x_{\text{LCF}} - y_{\text{LCF}} - x_{\text{Eff}} + y_{\text{Eff}}} \begin{pmatrix} x_{\text{LCF}} - y_{\text{LCF}} \\ -(x_{\text{Eff}} - y_{\text{Eff}}) \end{pmatrix} \tag{6.33}$$

Thus, ω_{Eff} is then calculated by

$$\omega_{\text{Eff}} = \frac{x_{\text{LCF}} - y_{\text{LCF}}}{x_{\text{LCF}} - y_{\text{LCF}} - x_{\text{Eff}} + y_{\text{Eff}}} \tag{6.34}$$

Note that this scheme is usually initialized with two non-dominated points which minimize the first and the second individual objective, respectively. Ideally, optimizing with respect to the new weight yields a new solution that can then be used to successively determine further weights.

In our case, we use the objective values of the solutions for $\omega_{\text{Eff}} \in \{0, 1\}$, see Figure 6.13a. These solutions do not not necessarily correspond to non-dominated points, but at the moment we do not have further information about relevant weight values for the optimization since the objective values differ too much in the order of magnitude. The first weight is then calculated by

$$\omega_{\text{Eff}} \approx \frac{2.80 \cdot 10^{-7} - 2.51 \cdot 10^{-9}}{2.80 \cdot 10^{-7} - 2.51 \cdot 10^{-9} - 557844.26 + 575523.92} \tag{6.35}$$

and we obtain $\omega_{\text{Eff}} = 1.57185136 \cdot 10^{-11}$.

The non-rounded values and results are shown in Table 6.3. The different runs are numbered consecutively. In addition, we have defined levels for calculating the new weighs based on the results already calculated. Given new points in the objective space further weights can successively be calculated. The concept of the method is illustrated in Figures 6.13a to 6.13d based on the calculated results. The connecting

With weight $\omega_{\text{Eff}} = 0$ and weight $\omega_{\text{LCF}} = 1$				
Iter.	J_{Eff}	$J_{\text{Eff}}/J_{\text{Eff}}(\Omega^{(0)})$	J_{LCF}	$J_{\text{LCF}}/J_{\text{LCF}}(\Omega^{(0)})$
0	564354.5	1.000	$1.6 \cdot 10^{-8}$	1.0
1	567428.4	1.005	$1.1 \cdot 10^{-8}$	0.70
2	570077.5	1.010	$8.6 \cdot 10^{-9}$	0.52
3	572252.7	1.014	$6.5 \cdot 10^{-9}$	0.40
4	575195.0	1.019	$5.1 \cdot 10^{-9}$	0.31
5	572880.9	1.015	$4.2 \cdot 10^{-9}$	0.25
6	572569.2	1.015	$3.3 \cdot 10^{-9}$	0.20
7	575061.7	1.019	$2.6 \cdot 10^{-9}$	0.16
8	575520.3	1.020	$2.5 \cdot 10^{-9}$	0.15
9	575523.9	1.020	$2.5 \cdot 10^{-9}$	0.15
With weight $\omega_{\text{Eff}} = 1$ and weight $\omega_{\text{LCF}} = 0$				
Iter.	J_{Eff}	$J_{\text{Eff}}/J_{\text{Eff}}(\Omega^{(0)})$	J_{LCF}	$J_{\text{LCF}}/J_{\text{LCF}}(\Omega^{(0)})$
0	564354.511	1	$1.632 \cdot 10^{-8}$	1
1	561312.744	0.9946	$2.413 \cdot 10^{-8}$	1.47
2	559489.848	0.9914	$4.101 \cdot 10^{-8}$	2.51
3	558276.945	0.9892	$9.273 \cdot 10^{-8}$	5.68
4	557844.261	0.9885	$2.804 \cdot 10^{-7}$	17.18

Table 6.2: Single-objective optimization (rounded values).

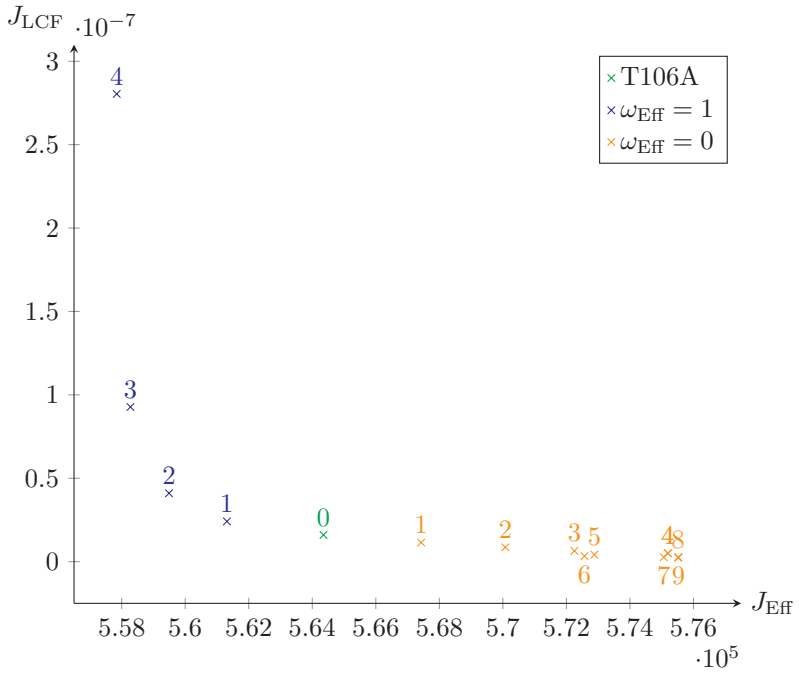
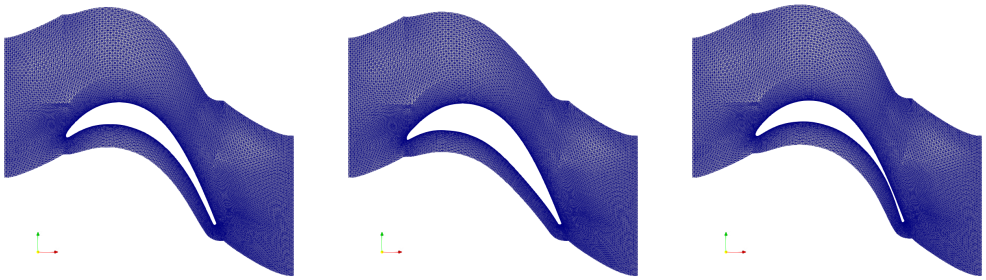


Figure 6.11: Objective values of iteration process for $\omega_{\text{Eff}} = 0$ and $\omega_{\text{Eff}} = 1$.



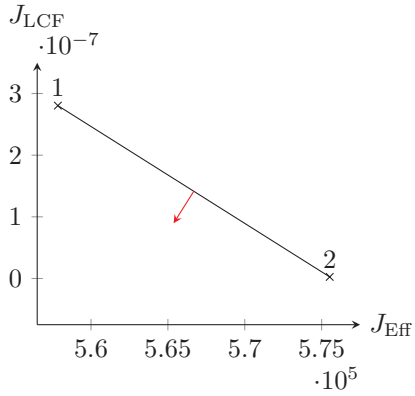
(a) Initial shape T106A. (b) Finial shape for $\omega_{\text{Eff}} = 0$. (c) Finial shape for $\omega_{\text{Eff}} = 1$.

Figure 6.12: Initial shape and final shapes for $\omega_{\text{Eff}} \in \{0, 1\}$.

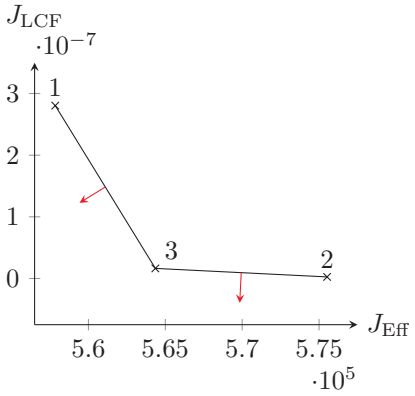
lines represents the new weights as in Section 2.2.1 explained, see in particular Figure 2.4. We proceed in *levels*, where the algorithm is executed with the new weights in each level before proceeding with the calculation of further weights. In Figures 6.13a to 6.13d, the solid connecting lines represent the new relevant weights of the current level, while the dashed lines represent weights that are not longer considered.

We can see that for the first weight the weighted sum optimization run number 3 terminates in iteration 0, see Table 6.3. Thus, no shape update was performed and the new point is given be the initial objective values. This can be the case if the initial shape is already efficient for the chosen weight, or if we terminate in a local minimum. Since both run number 3 and 4 get stuck at iteration 0, we do not get a new point by run number 4. Thus at level 3, only two new weights instead of four can be calculated. At level 4, we see that a negative weight for run number 8 was calculated by the results of run number 5 and 6. This is because the result of number 5 is dominated by the result of number 6, see Table 6.3. Thus, the result of run number 6 would also be a better solution for the weight of number 5. Since we do not want to evaluate negative weights, dominated points should be excluded for further weight calculations.

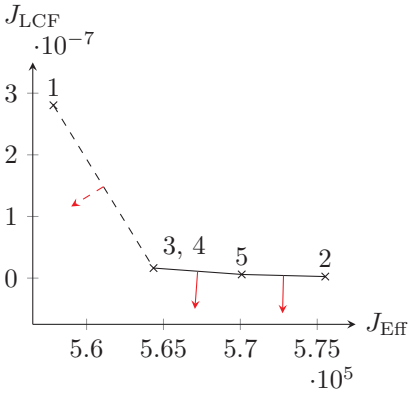
All in all, we see that the dichotomic search scheme calculated some new shapes improving the LCF objective while increasing the efficiency objective, see Figure 6.13e. Furthermore, we note that the algorithm can also get stuck in local minima or due to other numerical problems as outline above, since we have calculated two dominated points (run number 5 and 10 denoted by the red crosses in Figure 6.13e) and we were not able to calculate shapes that improve the efficiency objective with the chosen starting points of run number 1 ($\omega_{\text{Eff}} = 1$) and run number 2 ($\omega_{\text{Eff}} = 0$).



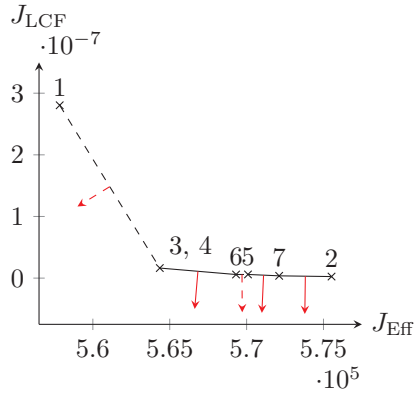
(a) Level 1.



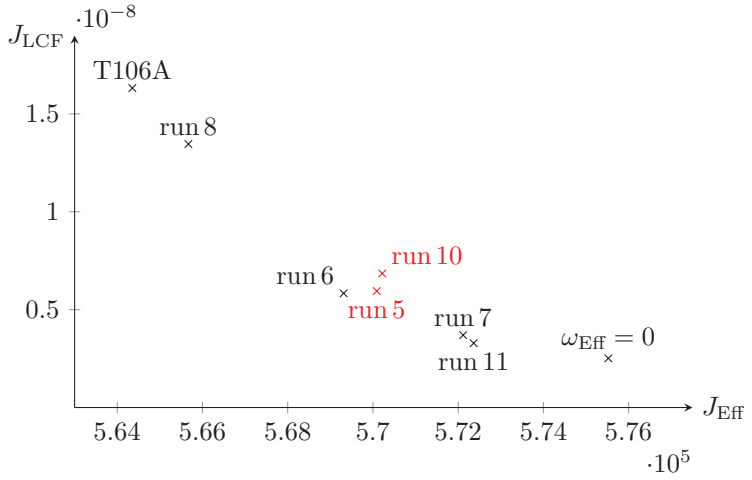
(b) Level 2.



(c) Level 3.



(d) Level 4.



(e) Results of run number 5 to 11 (zoomed in).

Figure 6.13: Dichotomic search scheme.

Level	Run	Calc. by	ω_{Eff}	J_{Eff}	J_{LCF}	Iter.
initial	1		1.0	557844.260731	$2.80411733306649 \cdot 10^{-7}$	4
points	2		0.0	575523.917708	$2.51380419155106 \cdot 10^{-9}$	9
1	3	(1, 2)	$1.57185136 \cdot 10^{-11}$	564354.511214	$1.63180454366424 \cdot 10^{-8}$	0
2	4	(1, 3)	$4.05658259 \cdot 10^{-11}$	564354.511214	$1.63180454366424 \cdot 10^{-8}$	0
	5	(3, 2)	$1.23589747 \cdot 10^{-12}$	570089.698148	$5.94679594553834 \cdot 10^{-9}$	5
3	6	(3, 5)	$1.80835422 \cdot 10^{-12}$	569312.7789	$5.83455775579977 \cdot 10^{-9}$	7
	7	(5, 2)	$6.31735931 \cdot 10^{-13}$	572117.313454	$3.69765979322555 \cdot 10^{-9}$	7
4	8	(3, 6)	$2.11434484 \cdot 10^{-12}$	565670.068729	$1.34575665413941 \cdot 10^{-8}$	2
	9	(6, 5)	$-1.44465709 \cdot 10^{-13}$			
	10	(5, 7)	$1.10925191 \cdot 10^{-12}$	570218.898775	$6.84845592043746 \cdot 10^{-9}$	4
	11	(7, 2)	$3.47517796 \cdot 10^{-13}$	572363.652626	$3.28527498434504 \cdot 10^{-9}$	7

Table 6.3: Dichotomic search results.

Choice of further weights Since we have not yet achieved any improvement in the efficiency objective (except with $\omega_{\text{Eff}} = 1$), we now focus on increasing the weight ω_{Eff}. The highest weight calculated by the performed dichotomic search scheme was approximately $1.57 \cdot 10^{-11}$. Therefore, we test the algorithm with weight $\omega_{\text{Eff}} = 10^{-\alpha}$ with $\alpha \in \{10, 9, 8, \ldots, 1\}$.

We note that for the weights $\omega_{\text{Eff}} \in \{10^{-10}, 10^{-9}, 10^{-8}, 10^{-7}, 10^{-6}\}$ the algorithm does not find a shape with the calculated search direction that improves the weighted sum scalarization. For the other weights $\alpha \leq 5$, the algorithm is able to perform some shape updates, see Figure 6.14. However, for the weights $\alpha = 1, 2, 3$ we realize that the algorithm quickly approaches infeasible shapes and terminates as the shapes get too thin and further shape updates lead to overlapping boundaries. In addition, the results obtained by $\alpha = 4, 5$ dominate the results obtained by $\alpha = 1, 2, 3$ and $\omega_{\text{Eff}} = 1$. From this we conclude that a relevant weight space could lie between $\omega_{\text{Eff}} = 10^{-5}$ and $\omega_{\text{Eff}} = 10^{-4}$. Therefore, we test the weights $\omega_{\text{Eff}} = \gamma \cdot 10^{-5}$ with $\gamma = 2, \ldots, 9$

The results are shown in Figure 6.15 together with the results of the dichtomous search scheme. For some outcome vectors, we have added the corresponding shape. The test runs with $10^{-5} < \omega_{\text{Eff}} < 10^{-4}$ were able to calculate some new solutions, but there are also some dominated solutions among them. The number of required iterations is 3 to 7. Previously non-dominated solutions are achieved by $\omega_{\text{Eff}} = 3 \cdot 10^{-5}$ and $\omega_{\text{Eff}} = 4 \cdot 10^{-5}$.

All in all, with this choice of weights we are able to calculate a few more solutions that improve the efficiency objective. The trade-off between reliability and efficiency is clearly recognizable. In addition, the LCF objective seems to have a regulating effect on the efficiency objective, as it prevents the algorithm from quickly generating

infeasible shapes where the profile becomes very thin at a certain point.

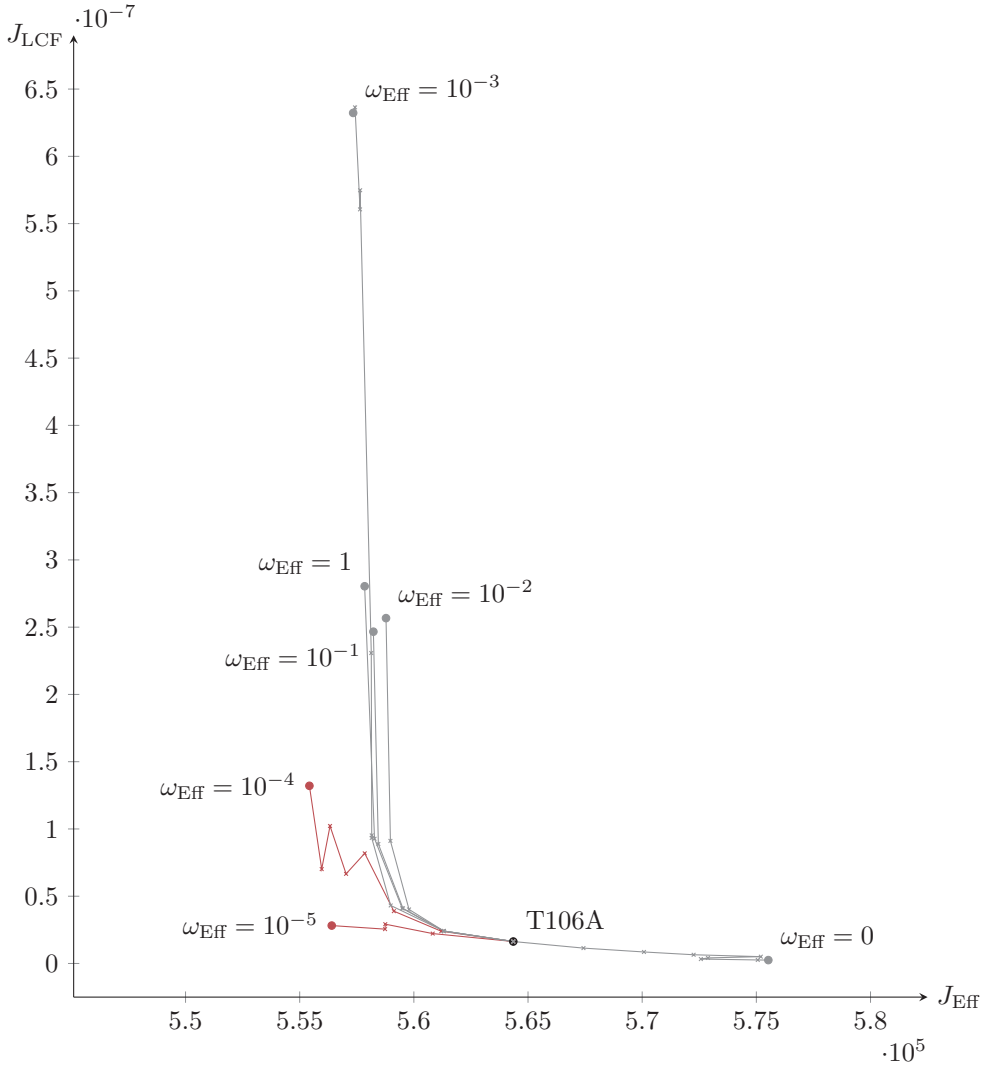


Figure 6.14: Objective values with iteration history for $\omega_{\mathrm{Eff}} = 10^{-\alpha}$ with $\alpha \in \{1, 2, 3, 4, 5\}$ and $\omega_{\mathrm{Eff}} \in \{0, 1\}$.

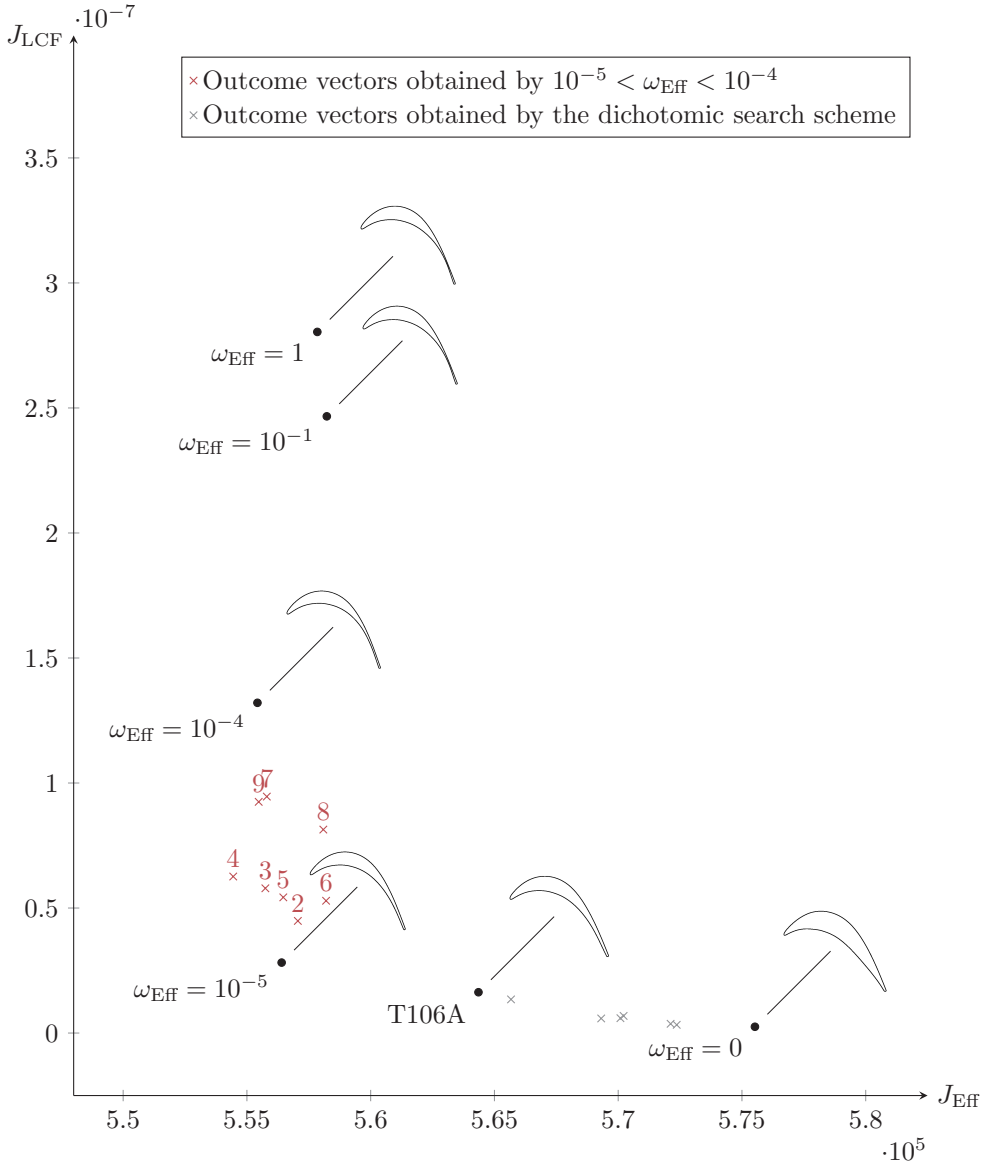


Figure 6.15: Obtained results.

Chapter 7

Conclusion

This thesis explores the integration of gradient based biobjective optimization methods for shape optimization problems using the weighted sum method and a hypervolume scalarization technique.

In Chapter 3 we present the theoretical framework of the hypervolume scalarization technique. Extending prior findings from Schultes et al., 2021 we generalize these results from the biobjective case to the multiobjective case. We study the behavior of the contour lines of the hypervolume indicator and show that they are concave. We point out that contour lines near the reference point resemble the contour lines of a weighted Chebyshev scalarization whereas the contour lines further away gradually approach the contour line of the weighted sum scalarization with equal weights. With these properties the hypervolume scalarization technique is not limited to calculate solely supported efficient solutions as the weighted sum method is. This applies also to the case that the chosen reference point is not already non-dominated. The hypervolume scalarization technique is well-suited for gradient based optimization as the gradient of the hypervolume indicator can easily be computed using the gradients of the primary objective functions, and does not add further constraints to the optimization problem with an appropriately chosen reference point.

Beyond these theoretical investigations, a substantial part of this thesis is dedicated to the implementation and application of scalarization methods to biobjective shape optimization problems. For this purpose, we select two distinct PDE constrained shape optimization problems as case studies.

In Chapter 5 we commence with the optimization of reliability and cost of ceramic components subjected to tensile load. The reliability objective is modeled as probability

of failure, while for the second objective we consider the volume of the component addressing its material cost. We develop two different implementations: one employing the weighted sum scalarization utilizing a parameter-based representation of the component via B-splines, and the other using the hypervolume scalarization with a parameter-free representation through structured grids. Both implementations yield successful approximations of the Pareto front.

Building upon these findings, we tackle a more complex optimization problem in Chapter 6. We consider the reliability and efficiency of the turbomachinery components selecting the low-pressure turbine cascade test case T106A. We use TRACE, a tool developed by the DLR, to assess the efficiency and model the reliability objective by low cycle fatigue. The challenge here is that both objectives depend on a different PDE, but the PDEs are not solved on the same domain. While the efficiency objectives depends on the Navier-Stokes equations calculated on the exterior domain of the component, the reliability objective depends on the linear elasticity equation calculated in the inside domain. We set up a coupling routine combing both objectives in an iterative line search solver and connect the sensitivities over the edge of the component. In addition, we use Steklov-Poincaré gradients in order to define a search direction such that we have a smooth deformation of the shape iterates. With the weighted sum scalarization method we successfully calculate a trade-off between both objectives.

Since the coupling of two finite element meshes is indeed a major challenge, irregular shapes or infeasible shape updates may lead to a premature and unsuccessful termination of the algorithm. Besides the reason that we may be in a local optimum for the current weighting, it may be the case that we have, for example, some numerical difficulties concerning the PDE calculations. Moreover, a possible cause can be that the objective values differ numerically too much in the order of magnitude, leading to catastrophic cancellation. However, it may also indicate that there is a scaling error in the sensitivities of the objective values such that the weighted sum gradient is not calculated correctly.

In addition to a further revision of the model, it would be worthwhile to perform the algorithm with other initial shapes. The initial shape chosen here, the T106A configuration, has already been thoroughly optimized, so that it may well be efficient or a local minimum. We would avoid that the algorithm gets stuck directly in the first iteration for some weights by choosing other stating shapes. It might also be possible to get into other areas of the Pareto front in this way. Furthermore, it would be useful to evaluate alternative methods, like the hypervolume scalarization, see Algorithm 5,

or multiobjective descent algorithm, see Algorithm 2. In this way, we could examine what other methods are able to calculate and determine if they encounter similar problems. Last but not least, it would also be interesting to include other objectives implemented in the TRACE tool in the optimization such as velocity angles, for example.

It is particularly noteworthy that we have succeeded in combining both objectives and thus creating a solid basis for further optimization using gradient based methods. With this method, our considerations go beyond pure efficiency calculations and include essential reliability criteria. This extension enriches the analysis and decision-making process, especially in the area of shape optimization for gas turbines.

Bibliography

Allaire, G. (2002). *Shape Optimization by the Homogenization Method*. Ed. by S. Antman, J. Marsden, and L. Sirovich. Vol. 146. Applied Mathematical Sciences. Springer New York. DOI: 10.1007/978-1-4684-9286-6.

Allaire, G. and F. Jouve (2008). "Minimum stress optimal design with the level set method". In: *Engineering Analysis with Boundary Elements* 32.11, pp. 909–918. DOI: 10.1016/j.enganabound.2007.05.007.

Alnæs, M. et al. (2015). "The FEniCS Project Version 1.5". en. In: *Archive of Numerical Software* 3.100. DOI: 10.11588/ANS.2015.100.20553.

Aneja, Y. P. and K. P. K. Nair (1979). "Bicriteria Transportation Problem". In: *Management Science* 25.1, pp. 73–78. ISSN: 1526-5501. DOI: 10.1287/mnsc.25.1.73.

Auger, A. et al. (2009). "Theory of the hypervolume indicator: optimal μ-distributions and the choice of the reference point". In: *Proceedings of the tenth ACM SIGEVO workshop on Foundations of genetic algorithms - FOGA '09*. ACM Press. DOI: 10.1145/1527125.1527138.

— (2012). "Hypervolume-based multiobjective optimization: Theoretical foundations and practical implications". In: *Theoretical Computer Science* 425, pp. 75–103. DOI: 10.1016/j.tcs.2011.03.012.

Babuška, I. and A. K. Aziz (1976). "On the Angle Condition in the Finite Element Method". In: *SIAM Journal on Numerical Analysis* 13.2, pp. 214–226. DOI: 10.1137/0713021.

Backhaus, J. (2020). "Adjungierte Strömungssimulation und gradientenbasierte Ersatzmodelle in der Turbomaschinenauslegung". de. PhD thesis. DOI: 10.13154/294-7557.

Backhaus, J., M. Bolten, et al. (2021). "GivEn—Shape Optimization for Gas Turbines in Volatile Energy Networks". In: *Mathematical Modeling, Simulation and Opti-*

mization for Power Engineering and Management. Ed. by S. Göttlich, M. Herty, and A. Milde. Vol. 34. Mathematics in Industry. Cham: Springer, pp. 71–106. DOI: 10.1007/978-3-030-62732-4_4.

Backhaus, J., A. Schmitz, et al. (2017). "Application of an Algorithmically Differentiated Turbomachinery Flow Solver to the Optimization of a Fan Stage". In: *18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2017-3997.

Balay, S. et al. (2019). *PETSc Users Manual*. Argonne National Laboratory.

Bazaraa, M. S. (2006). *Nonlinear programming. theory and algorithms*. J. Wiley & Sons. ISBN: 0471486000.

Becker, K., K. Heitkamp, and E. Kügeler (2010). "Recent Progress In A Hybrid-Grid CFD Solver For Turbomachinery Flows". In: *Proceedings Fifth European Conference on Computational Fluid Dynamics ECCOMAS CFD 2010*. Ed. by J. C. F. Pereira, A. Sequeira, and J. M. C. Pereira. URL: https://elib.dlr.de/68938/.

Beume, N. et al. (2009). "Effects of 1-Greedy $\mathcal{S}$-Metric-Selection on Innumerably Large Pareto Fronts". In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 21–35. DOI: 10.1007/978-3-642-01020-0_7.

Bhimasankaram, P. and A. Rao (2000). *Linear Algebra*. Springer-Verlag GmbH. 428 pp. ISBN: 9789386279019. URL: https://www.ebook.de/de/product/35119946/p_bhimasankaram_a_ramachandra_rao_linear_algebra.html.

Bittner, L. (2018). "On Shape Calculus with Elliptic PDE Constraints in Classical Function Spaces". PhD thesis. DOI: 10.25926/PKFR-KR55.

Bolten, M., O. T. Doganay, et al. (2024). "Non-convex shape optimization by dissipative Hamiltonian flows". In: *Engineering Optimization*, pp. 1–20. ISSN: 1029-0273. DOI: 10.1080/0305215x.2024.2304135.

Bolten, M., H. Gottschalk, C. Hahn, et al. (2019). "Numerical shape optimization to decrease failure probability of ceramic structures". In: *Comput. Visual Sci.* 21, pp. 1–10. DOI: 10.1007/s00791-019-00315-z.

Bolten, M., H. Gottschalk, and S. Schmitz (2015). "Minimal Failure Probability for Ceramic Design Via Shape Control". In: *J Optim Theory Appl* 166.3, pp. 983–1001. DOI: 10.1007/s10957-014-0621-8.

Bons, N. P. et al. (2019). "Multimodality in Aerodynamic Wing Design Optimization". In: *AIAA Journal* 57.3, pp. 1004–1018. DOI: 10.2514/1.j057294.

Bowman, V. J. (1976). "On the Relationship of the Tchebycheff Norm and the Efficient Frontier of Multiple-Criteria Objectives". In: *Lecture Notes in Economics and*

Mathematical Systems. Springer Berlin Heidelberg, pp. 76–86. DOI: 10.1007/978-3-642-87563-2_5.

Braess, D. (2013). *Finite Elemente*. Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-34797-9.

Bringmann, K., S. Cabello, and M. Emmerich (2017). "Maximum Volume Subset Selection for Anchored Boxes". en. In: DOI: 10.4230/LIPICS.SOCG.2017.22.

Bringmann, K., T. Friedrich, and P. Klitzke (2014). "Two-dimensional subset selection for hypervolume and epsilon-indicator". In: *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*. ACM. DOI: 10.1145/2576768.2598276.

Brückner-Foit, A. et al. (1997). "Discrimination of multiaxiality criteria with the Brazilian disc test". In: *Journal of the European Ceramic Society* 17.5, pp. 689–696. DOI: 10.1016/s0955-2219(96)00085-4.

Bucur, D. and G. Buttazzo (2005). *Variational Methods in Shape Optimization Problems*. Birkhäuser Boston. DOI: 10.1007/b137163.

Büsing, C. et al. (2017). "Reference points and approximation algorithms in multicriteria discrete optimization". In: *European Journal of Operational Research* 260.3, pp. 829–840. DOI: 10.1016/j.ejor.2016.05.027.

Chenais, D. (1975). "On the existence of a solution in a domain identification problem". In: *Journal of Mathematical Analysis and Applications* 52.2, pp. 189–219. DOI: 10.1016/0022-247x(75)90091-8.

Chirkov, D. V. et al. (2018). "Multi-objective shape optimization of a hydraulic turbine runner using efficiency, strength and weight criteria". In: *Struct Multidisc Optim* 58.2, pp. 627–640. DOI: 10.1007/s00158-018-1914-6.

Cohon, J. L. (1978). *Multiobjective programming and planning*. Academic Press, p. 333. ISBN: 0121783502.

Conti, S. et al. (2009). "Shape Optimization Under Uncertainty—A Stochastic Programming Perspective". In: *SIAM J. Optim.* 19.4, pp. 1610–1632. DOI: 10.1137/070702059.

Dandurand, B. and M. M. Wiecek (2016). "Quadratic scalarization for decomposed multiobjective optimization". In: *OR Spectrum* 38.4, pp. 1071–1096. DOI: 10.1007/s00291-016-0453-z.

Das, I. and J. E. Dennis (1997). "A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems". In: *Structural Optimization* 14.1, pp. 63–69. DOI: 10.1007/bf01197559.

Deb (2001). *Multi-Objective Optimization*. John Wiley & Sons. 536 pp. ISBN: 047187339X. URL: https://www.ebook.de/de/product/5833814/deb_multi_objective_optimization.html.

Deb, K. (2011). "Multi-objective Optimisation Using Evolutionary Algorithms: An Introduction". In: *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*. Ed. by L. Wang, A. H. C. Ng, and K. Deb. Springer London, pp. 3–34. ISBN: 978-0-85729-652-8. DOI: 10.1007/978-0-85729-652-8_1.

Deb, K. and T. Goel (2002). "Multi-Objective Evolutionary Algorithms for Engineering Shape Design". In: *Evolutionary Optimization*. Vol. 48. International Series in Operations Research & Management Science. Boston, MA: Springer US, pp. 147–175. ISBN: 978-0-306-48041-6. DOI: 10.1007/0-306-48041-7_6.

Delfour, M. C. and J. .-.-P. Zolésio (2011). *Shapes and Geometries*. 2nd ed. Advances in Design and Control. Society for Industrial and Applied Mathematics. ISBN: 978-0-898719-36-9. DOI: 10.1137/1.9780898719826.

Désidéri, J.-A. (2009). *Multiple-Gradient Descent Algorithm (MGDA)*. Research rep. RR-6953. INRIA. URL: https://hal.inria.fr/inria-00389811.

— (2012). "Multiple-gradient descent algorithm (MGDA) for multiobjective optimization". In: *Comptes Rendus Mathematique* 350.5-6, pp. 313–318. DOI: 10.1016/j.crma.2012.03.014.

Dietz, H. M. (2019). *Mathematik für Wirtschaftswissenschaftler*. Springer Berlin Heidelberg. DOI: 10.1007/978-3-662-58149-0.

Doganay, O. T. et al. (2020). "Gradient based biobjective shape optimization to improve reliability and cost of ceramic components". In: *Optim Eng* 21.4, pp. 1359–1387. DOI: 10.1007/s11081-019-09478-7.

Duran, R. G. and M. A. Muschietti (2004). "The Korn inequality for Jones domains". English. In: *Electron. J. Differ. Equ.* 2004.127, p. 10. ISSN: 1072-6691. URL: https://ejde.math.txstate.edu/Volumes/2004/127/duran.pdf.

Duysinx, P. and M. P. Bendsøe (1998). "Topology optimization of continuum structures with local stress constraints". In: *Int. J. Numer. Meth. Engng.* 43.8, pp. 1453–1478. DOI: 10.1002/(sici)1097-0207(19981230)43:8<1453::aid-nme480>3.0.co;2-2.

Ehrgott, M. (2005). *Multicriteria Optimization*. 2nd ed. Springer Berlin Heidelberg. 340 pp. ISBN: 978-3-540-21398-7. DOI: 10.1007/3-540-27659-9.

Emmerich, M. and A. Deutz (2014). "Time Complexity and Zeros of the Hypervolume Indicator Gradient Field". In: *EVOLVE - A Bridge between Probability, Set*

Oriented Numerics, and Evolutionary Computation III. Springer International Publishing, pp. 169–193. DOI: 10.1007/978-3-319-01460-9_8.

Eppler, K. (2017). "On Hadamard shape gradient representations in linear elasticity". Unpublished manuscript.

Eppler, K., H. Harbrecht, and R. Schneider (2007). "On Convergence in Elliptic Shape Optimization". In: *SIAM J. Control Optim.* 46.1, pp. 61–83. DOI: 10.1137/05062679x.

Fliege, J., A. I. F. Vaz, and L. N. Vicente (2019). "Complexity of gradient descent for multiobjective optimization". In: *Optimization Methods and Software* 34.5, pp. 949–959. DOI: 10.1080/10556788.2018.1510928. eprint: https://doi.org/10.1080/10556788.2018.1510928. URL: https://doi.org/10.1080/10556788.2018.1510928.

Fliege, J. (2004). "Gap-free computation of Pareto-points by quadratic scalarizations". In: *Mathematical Methods of Operations Research (ZOR)* 59.1, pp. 69–89. DOI: 10.1007/s001860300316.

Fliege, J. and B. F. Svaiter (2000). "Steepest descent methods for multicriteria optimization". In: *Mathematical Methods of OR* 51.3, pp. 479–494. DOI: 10.1007/s001860000043.

Freimer, M. and P. L. Yu (1976). "Some New Results on Compromise Solutions for Group Decision Problems". In: *Management Science* 22.6, pp. 688–693. DOI: 10.1287/mnsc.22.6.688.

Fujii, N. (1988). "Lower semicontinuity in domain optimization problems". In: *J Optim Theory Appl* 59.3, pp. 407–422. DOI: 10.1007/bf00940307.

Geiger, C. and C. Kanzow (1999). *Numerische Verfahren zur Lösung unrestringierter Optimierungsaufgaben*. Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-58582-1.

Geoffrion, A. M. (1968). "Proper efficiency and the theory of vector maximization". In: *Journal of Mathematical Analysis and Applications* 22.3, pp. 618–630. ISSN: 0022-247X. DOI: 10.1016/0022-247x(68)90201-1. URL: https://www.sciencedirect.com/science/article/pii/0022247X68902011.

GE's HA Gas Turbine Delivers Second World Record for Efficiency (2018). Press Release. URL: https://www.ge.com/news/press-releases/ges-ha-gas-turbine-delivers-second-world-record-efficiency.

Giacomini, M., J.-A. Désidéri, and R. Duvigneau (2014). *Comparison of multiobjective gradient-based methods for structural shape optimization*. Tech. rep. RR-8511. INRIA. URL: https://hal.inria.fr/hal-00967601/document.

Giles, M. B. and N. A. Pierce (2000). "An Introduction to the Adjoint Approach to Design". In: *Flow, Turbulence and Combustion* 65.3/4, pp. 393–415. DOI: 10.1023/a:1011430410075.

Gottschalk, H. and M. Reese (2021). "An Analytical Study in Multi-physics and Multi-criteria Shape Optimization". In: *Journal of Optimization Theory and Applications* 189.2, pp. 486–512. DOI: 10.1007/s10957-021-01841-y.

Gottschalk, H. and M. Saadi (2019). "Shape gradients for the failure probability of a mechanic component under cyclic loading: a discrete adjoint approach". In: *Comput Mech* 64.4, pp. 895–915. DOI: 10.1007/s00466-019-01686-3.

Gottschalk, H., M. Saadi, et al. (2018). "Adjoint Method to Calculate the Shape Gradients of Failure Probabilities for Turbomachinery Components". In: *Proceedings of the ASME Turbo Expo 2018: Turbomachinery Technical Conference and Exposition*. Vol. 7A: Structures and Dynamics. V07AT32A003. Oslo, Norway. June 11-15: American Society of Mechanical Engineers. DOI: 10.1115/gt2018-75759.

Gottschalk, H. and S. Schmitz (2014). "Optimal Reliability in Design for Fatigue Life". In: *SIAM J. Control Optim.* 52.5, pp. 2727–2752. DOI: 10.1137/120897092.

Gottstein, G. (2004). *Physical Foundations of Materials Science*. Springer Berlin Heidelberg. DOI: 10.1007/978-3-662-09291-0.

Gross, D. and T. Seeling (2006). *Fracture Mechanics. With an introduction to micromechanics*. Mechanical Engineering Series. Berlin New York: Springer. ISBN: 3540240349. DOI: 10.1007/b22134.

Guerreiro, A. P. and C. M. Fonseca (2020). "An analysis of the Hypervolume Sharpe-Ratio Indicator". In: *European Journal of Operational Research* 283.2, pp. 614–629. DOI: 10.1016/j.ejor.2019.11.023.

Guerreiro, A. P., C. M. Fonseca, and L. Paquete (2016). "Greedy Hypervolume Subset Selection in Low Dimensions". In: *Evolutionary Computation* 24.3, pp. 521–544. DOI: 10.1162/evco_a_00188.

Hahn, C. (2021). "Auto-generated structured meshes for evolving domains". PhD thesis. DOI: 10.25926/GMYP-7S53.

Haslinger, J. and R. A. E. Mäkinen (2003). *Introduction to Shape Optimization*. Advances in Design and Control. Society for Industrial and Applied Mathematics. ISBN: 0-89871-536-9. DOI: 10.1137/1.9780898718690.

Hernandez, V. A. S. et al. (2020). "The Set-Based Hypervolume Newton Method for Bi-Objective Optimization". In: *IEEE Transactions on Cybernetics* 50.5, pp. 2186–2196. DOI: 10.1109/tcyb.2018.2885974.

Hoheisel, H. et al. (1986). "Influence of Free Stream Turbulence and Blade Pressure Gradient on Boundary Layer and Loss Behaviour of Turbine Cascades". In: *Volume 1: Turbomachinery*. American Society of Mechanical Engineers. DOI: 10.1115/86-gt-234.

Kallenberg, O. (1983). *Random measures*. English. 3rd ed. Berlin: Akademie-Verlag.

Kuhn, T. et al. (2016). "Hypervolume Subset Selection in Two Dimensions: Formulations and Algorithms". In: *Evolutionary Computation* 24.3, pp. 411–425. DOI: 10.1162/evco_a_00157.

Laurain, A. and K. Sturm (2016). "Distributed shape derivative *via* averaged adjoint method and applications". In: *ESAIM: M2AN* 50.4, pp. 1241–1267. DOI: 10.1051/m2an/2015075.

Les Piegl, W. T. (2000). *The NURBS book. Monographs in visual communication*. Springer Berlin Heidelberg.

Liesen, J. and V. Mehrmann (2021). *Lineare Algebra*. Springer Berlin Heidelberg. 386 pp.

Logg, A. and G. N. Wells (2010). "DOLFIN". In: *ACM Transactions on Mathematical Software* 37.2, pp. 1–28. DOI: 10.1145/1731022.1731030.

Luft, D. (2021). "Pre-Shape Calculus - a Unified Framework for Mesh Quality and Shape Optimization". en. PhD thesis. DOI: 10.25353/UBTR-XXXX-63FE-D751.

Michor, P. and D. Mumford (2006). "Riemannian geometries on spaces of plane curves". In: *Journal of the European Mathematical Society*, pp. 1–48. DOI: 10.4171/jems/37.

Miettinen, K. (1998). *Nonlinear Multiobjective Optimization*. International Series in Operations Research & Management Science. Springer US. DOI: 10.1007/978-1-4615-5563-6.

Miettinen, K. and M. M. Mäkelä (1996). "NIMBUS — Interactive Method for Nondifferentiable Multiobjective Optimization Problems". In: *Multi-Objective Programming and Goal Programming*. Springer Berlin Heidelberg, pp. 50–57. DOI: 10.1007/978-3-642-87561-8_5.

— (2002). "On scalarizing functions in multiobjective optimization". In: *OR Spectrum* 24.2, pp. 193–213. DOI: 10.1007/s00291-001-0092-9.

Morell, R. (2004). *Brevier technical ceramics*. Tech. rep. Verband der Keramischen Industrie e.V., Informationszentrum Technical Ceramics. URL: http://www.keramverband.de/brevier_engl/brevier.htm.

Morsbach, C. (2017). "Reynolds Stress Modelling for Turbomachinery Flow Applications". PhD thesis. DLR. URL: https://elib.dlr.de/113258/.

Munz, D. and T. Fett (2001). *Ceramics. Mechanical Properties, Failure Behaviour, Materials Selection*. Ed. by R. Hull et al. 2nd ed. Vol. 36. Springer Series in Materials Science. Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-58407-7.

Neuber, H. (1961). "Theory of Stress Concentration for Shear-Strained Prismatical Bodies With Arbitrary Nonlinear Stress-Strain Law". In: *Journal of Applied Mechanics* 28.4, pp. 544–550. ISSN: 1528-9036. DOI: 10.1115/1.3641780.

OMG (2011). *Business Process Model and Notation (BPMN), Version 2.0*. Object Management Group. URL: http://www.omg.org/spec/BPMN/2.0.

Picelli, R. et al. (2018). "Stress-based shape and topology optimization with the level set method". In: *Comput. Methods Appl. Mech. Engrg.* 329, pp. 1–23. DOI: 10.1016/j.cma.2017.09.001.

Przybylski, A., K. Klamroth, and R. Lacour (2019). *A simple and efficient dichotomic search algorithm for multi-objective mixed integer linear programs*. DOI: 10.48550/ARXIV.1911.08937.

Pulliam, T. et al. (2003). "Comparison of Evolutionary (Genetic) Algorithm and Adjoint Methods for Multi-Objective Viscous Airfoil Optimizations". In: *41st Aerospace Sciences Meeting and Exhibit*. 6-9 January 2003, Reno, Nevada: American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2003-298.

Radaj, D. and M. Vormwald (2007). *Ermüdungsfestigkeit*. Springer Berlin Heidelberg. DOI: 10.1007/978-3-540-71459-0.

Ramberg, W. and W. Osgood (1943). *Description of stress-strain curves by three parameters*. Tech. rep. Technical note 902, National Advisory Committee for Aeronautics, Washington, DC.

Rösler, J., H. Harders, and M. Bäker (2019). *Mechanisches Verhalten der Werkstoffe*. 6th ed. Springer Fachmedien Wiesbaden. DOI: 10.1007/978-3-658-26802-2.

Roudi, S., H. Riesch-Oppermann, and O. Kraft (2005). "Advanced probabilistic tools for the uncertainty assessment in failure and lifetime prediction of ceramic components". In: *Materialwissenschaft und Werkstofftechnik* 36.3-4, pp. 171–176. DOI: 10.1002/mawe.200500861.

Saadi, M. (2021). “Shape Sensitivities for the Failure Probability of Mechanical Components”. PhD thesis. DOI: 10.25926/DP9R-HJ27.

Sachverständigenrat für Umweltfragen (2011). *Wege zur 100% erneuerbaren Stromversorgung. Sondergutachten, Januar 2011*. Erich Schmidt Verlag. ISBN: 9783503136063. URL: https://www.umweltrat.de/SharedDocs/Downloads/DE/02_Sondergutachten/2008_2012/2011_07_SG_Wege_zur_100_Prozent_erneuerbaren_Stromversorgung.pdf.

Sagebaum, M. et al. (2017). “Efficient Algorithmic Differentiation Techniques for Turbo-machinery Design”. In: *18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2017-3998.

Schlömer, N. (2019). *meshio: Tools for mesh files*. URL: https://github.com/nschloe/meshio.

Schmidt, S. and V. Schulz (2009). “Impulse Response Approximations of Discrete Shape Hessians with Application in CFD”. In: *SIAM Journal on Control and Optimization* 48.4, pp. 2562–2580. DOI: 10.1137/080719844.

Schmitz, S. (2014). *A local and probabilistic model for low-cycle fatigue new aspects of structural analysis*. Konstanz: Hartung-Gorre. ISBN: 9783866285118.

Schmitz, S., H. Gottschalk, et al. (2013). “Risk Estimation for LCF Crack Initiation”. In: *Volume 7A: Structures and Dynamics*. American Society of Mechanical Engineers. DOI: 10.1115/gt2013-94899.

Schmitz, S., G. Rollmann, et al. (2013). “Probabilistic Analysis of LCF Crack Initiation Life of a Turbine Blade under Thermomechanical Loading”. In: Conference: LCF7, Seventh International Conference on Low Cycle Fatigue. arXiv: 1310.0629 [math.NA].

Schmitz, S., T. Seibel, et al. (2013). “A probabilistic model for LCF”. In: *Computational Materials Science* 79, pp. 584–590. DOI: 10.1016/j.commatsci.2013.07.015.

Schultes, J. et al. (2021). “Hypervolume scalarization for shape optimization to improve reliability and cost of ceramic components”. In: *Optim Eng* 22.2, pp. 1203–1231. DOI: 10.1007/s11081-020-09586-9.

Schulz, V. (2014). “A Riemannian View on Shape Optimization”. In: *Found Comput Math* 14.3, pp. 483–501. DOI: 10.1007/s10208-014-9200-5.

Schulz, V. and M. Siebenborn (2016). “Computational Comparison of Surface Metrics for PDE Constrained Shape Optimization”. In: *Computational Methods in Applied Mathematics* 16.3, pp. 485–496. DOI: 10.1515/cmam-2016-0009.

Schulz, V., M. Siebenborn, and K. Welker (2016). "Efficient PDE Constrained Shape Optimization Based on Steklov–Poincaré-Type Metrics". In: *SIAM J. Optim.* 26.4, pp. 2800–2819. DOI: 10.1137/15m1029369.

Schulze, B. et al. (2020). "On the rectangular knapsack problem: approximation of a specific quadratic knapsack problem". In: *Mathematical Methods of Operations Research* 92.1, pp. 107–132. DOI: 10.1007/s00186-020-00702-0.

Shackelford, J. F. et al. (2015). *CRC Materials Science and Engineering Handbook*. Taylor & Francis Group, p. 634. ISBN: 9781482216530.

Sokolowski, J. and J.-P. Zolesio (1992). *Introduction to Shape Optimization. Shape Sensitivity Analysis*. Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-58106-9.

The Energy of the Future. 8th Monitoring Report on the Energy Transition –Reporting Years 2018 and 2019 (2021). Federal Ministry for Economic Affairs and Energy (BMWi). URL: https://www.bmwk.de/Redaktion/EN/Publikationen/Energie/the-energy-of-the-future-8th-monitoring-report.pdf.

Touré, C. et al. (2019). "Uncrowded hypervolume improvement". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM. DOI: 10.1145/3321707.3321852.

TRACE User Guide (2019). TRACE version 9.2.652. German Aerospace Center (DLR). URL: http://trace-portal.de/userguide/trace/index.html.

Tröltzsch, F. (2009). *Optimale Steuerung partieller Differentialgleichungen*. Vieweg+Teubner. DOI: 10.1007/978-3-8348-9357-4.

Watanabe, S. (1964). "On discontinuous additive functionals and Lévy measures of a Markov process". In: *Japanese journal of mathematics :transactions and abstracts* 34, pp. 53–70. DOI: 10.4099/jjm1924.34.0_53.

Weibull, W. (1939). *A Statistical Theory of the Strength of Materials*. Handlingar / Ingeniörsvetenskapsakademien. Generalstabens litografiska anstalts förlag. URL: https://books.google.de/books?id=otVRAQAAIAAJ.

Wilcox, D. C. (1988). "Reassessment of the scale-determining equation for advanced turbulence models". In: *AIAA Journal* 26.11, pp. 1299–1310. ISSN: 1533-385X. DOI: 10.2514/3.10041.

Yang, K. et al. (2019). "Efficient computation of expected hypervolume improvement using box decomposition algorithms". In: *Journal of Global Optimization* 75.1, pp. 3–34. DOI: 10.1007/s10898-019-00798-7.

Yu, P. L. (1973). “A Class of Solutions for Group Decision Problems”. In: *Management Science* 19.8, pp. 936–946. DOI: 10.1287/mnsc.19.8.936.

Zavala, G. R. et al. (2013). “A survey of multi-objective metaheuristics applied to structural optimization”. In: *Struct Multidisc Optim* 49.4, pp. 537–558. DOI: 10.1007/s00158-013-0996-4.

Zitzler, E. and L. Thiele (1999). “Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach”. In: *IEEE Transactions on Evolutionary Computation* 3.4, pp. 257–271. DOI: 10.1109/4235.797969.